J Ambient Intell Human Comput (2015) 6:647–659
DOI 10.1007/s12652-015-0281-z

ORIGINAL RESEARCH

The behavioral profiling based on times series forecasting for smart homes assistance

Mohamed Tarik Moutacalli[1] · **Abdenour Bouzouane**[1] · **Bruno Bouchard**[1]

Received: 31 October 2014 / Accepted: 7 April 2015 / Published online: 12 May 2015

Abstract The many disadvantages of traditional assistance available to elderly and persons with cognitive dysfunction such as patients with Alzheimer's disease have motivated the research of Technological assistance. The artificial agent, who will support the caregiver, is equipped with hardware and software resources that enable it to observe, analyze, infer and support, when needed, the assisted person. In this paper, we present the various stages of Technological assistance and propose a new algorithm for the step of activities models detection. We also explore an activity prediction step using time series. The experiments were conducted on real data recorded at LIARA smart home and the results are satisfactory.

Keywords Assisted living · Domestic care · Activity recognition · Activity detection · Activity prediction · Frequent pattern mining · Time series prediction · Smart homes

1 Introduction

According to (Alzheimer Society 2014) statistics, Canadians with Alzheimer disease are estimated to 500,000 persons and dementia care costs 15 billion dollars to Canadians. The majority of these expenditures are related to the traditional assistance where caregivers help them, when needed, accomplishing started activities and proposing the forgotten ones. Apart high costs, traditional assistance has several other disadvantages such as the loss of patient privacy, the difficult relationship that usually develops between the patient and the caregiver and accelerating the loss of patient autonomy who rely more on caregiver to accomplish his activities, etc (Dupuis et al. 2004). The emergence of ambient intelligence (AmI) (Sadri 2011), which is defined by a vision of a future in which environments support the people inhabiting them, has helped design a Technological assistance that, combined with traditional assistance, eliminates most of the disadvantages already mentioned. For good collaboration between the caregiver and the ambient agent, they should have the same abilities and work almost the same way. Therefore, the ambient agent must be able to perform the three steps of assistance: making observations, recognizing the activity being started and offering help when needed.

The first stage of the assistance process is a very important stage where all interesting observations are recorded. These observations are not only used for recognizing the patient started activity, but also for creating a user profile detailing daily living activities (ADLs) that he is used to perform by analysing the observations history log. To make and record observations, the patient house must be transformed to a smart house (Augusto and Nugent 2006) by equipping it by sensors capable of making and sending different measures. Several sensors may be used in a smart home, for example: electromagnetic contacts placed on doors, pressure sensors to detect the patient position, or radio-frequency identification (RFID) tags (Metras 2005) integrated into the miniature objects daily living (clothing, cups, dishes, etc.) to indicate the object position, etc. Existing research can be divided into two categories

✉ Mohamed Tarik Moutacalli
mohamed-tarik.moutacalli@uqac.ca

Abdenour Bouzouane
Abdenour_Bouzouane@uqac.ca

Bruno Bouchard
Bruno_Bouchard@uqac.ca

[1] Department of Informatics, UQAC, Chicoutimi, Canada

according to the type of observation used. In the first category, the observer agent observes directly the entity observed (the patient). For instance, Jalal et al. (2011) used cameras to record video clips which were analysed to train their system in order to recognize activities like cleaning, cooking, exercise, etc. Srinivasan et al. (2010) used a wearable sensor: Actigraph wrist watch to detect activities like cooking, sleep, eating, etc. Our research rather belongs to the second category where, as shown in Fig. 1, the observer agent observes environment changes and tries to infer the patient planned activity which preserves more the patient privacy. Table 1 schematizes the data warehouse obtained by recording all sensors measurements.

Before detailing the second stage, it should be noted that the third stage is conducted by equipping the smart house by effectors that propose help to the patient by sending a vocal on speakers, a video on television or using a more discrete media (light, emoticon, beep, etc.) (Van Tassel 2011). A comprehensive study on the patient health should be conducted to identify appropriate effectors to use.

Activity recognition is the most challenging stage in the process of technological assistance (Chen et al. 2012). It is a two steps stage; first, we detect all activities models, then we find among them the one that best explains the current activated sensors list. An activity model reflects the way in which the occupant is used to performing this activity. Since these activities models must allow the ambient agent to know the next sensor to be activated and to detect if the occupant is struggling to continue the activity, an activity

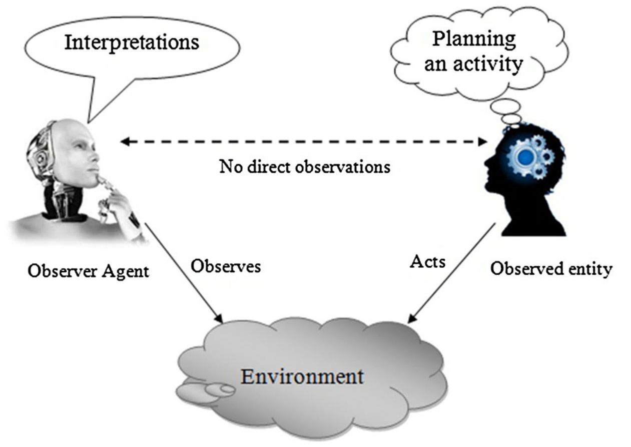

Fig. 1 Observation process

Table 1 All recorded sensors values

Date	Time	Sensor1	Sensor...	SensorN
01/03/2015	08:50:20.0	On	...	1245
01/03/2015	08:50:20.5	Off	...	1245

model consists of an ordered sequence of sensors that contains the estimated delay between two adjacent sensors. For example, this part of an activity model can be read:

$$\mathrm{CA9}(5-15), \mathrm{MLK}(1-4), \mathrm{CA9}(1-2)$$

The fridge door, represented by the tag-object CA9, will be activated in 5–15 s. After that, milk, represented by the tag-object MLK, will be activated in 1–4 s; and 1–2 s after the milk activation, the fridge door will be activated again.

Several problems may be encountered during the activity search process. The high number of daily living activities that a person is able to perform as well as the high number of activities models, can make this process complex and slow. Having m activities models, if we observe n actions, the execution time of finding the activity or activities that contains the observed actions is in $O(nm)$. Therefore, considering only the most likely activities will reduce the execution time. The second issue is an equiprobability problem. It occurs when several activities models contain the observed actions. In this case, all activities have the same probability of being carried out, and if the occupant is no longer able to complete its activity, the ambient agent would not know which one to propose. It is notable that the inability to initiate an activity prevents its recognition since no sensor will be activated. To address all cited problems, we propose an activities prediction step before the activity search stage. The following scenario highlights the role of activities start time prediction:

Suppose that we have sixty activities models and the activities start times predicted are: 0830 hours for Activity1, 0835 hours for Activity2, 0845 hours for Activity3 and the other activities are predicted after 0900 hours. At 0832 hours, for example, we can start looking, at first, for the activity among those predicted at a time close enough to the current time, i.e. to search among the first three activities instead of sixty. Furthermore, if current activated sensors belong to the first three activities, or if no sensor is activated, then Activity1 will most likely be performed since it is the closest to the current time.

As shown in the example, the prediction accuracy plays a key role in identifying the most likely activity the one that will be proposed to the home occupant if no action is detected. Two minutes later, if Activity2 predicted start time was 0833 hours, the most likely activity would be Activity2 instead of Activity1. Getting the most accurate prediction and introducing a seasonal effect in order to differentiate each week day explain our use of time series forecasting techniques.

Our contribution consists thus in detecting activities models and their start times from a sensor history log, then, in exploring time series techniques (Box et al. 1994) to predict activities start times in new horizons that will be used to predict the most likely activity to be performed.

The paper is organized as follows. We first discuss the existing approaches towards activities models detections and activity prediction. Then, our frequent pattern mining algorithm for activities models creation is summarised. Next, we detail the time series forecasting technique used in activities start times prediction and our activity prediction system. After that, we describe the experimental results of the approach employing real sensor database.

2 Related work

The assistance, in general, aims to offer help when irregular behavior is detected. In order to classify a behavior as irregular, regular behaviors must thus be defined. The many researches proposed for this area (Suryadevara et al. 2011, 2013; Jurek et al. 2014) don't only differ in irregularities detection approaches, but also in regular behaviors creation.

Regular behaviors may be created in a supervised way by incorporating a training phase where the home occupant performs separately all activities. For instance, in the training phase of Suryadevara et al. (2013) approach, the observed person was asked to fill, manually, a timesheet with activities whenever an activity has been performed. For detecting irregularities, they start by recognizing the started activity. They create an activities sequence AS composed with letters representing activities that contains each detected activated sensor. For example, if the first activated sensor belongs to the first and second activity and the second activated sensor belongs to the first and third activity, then: $AS = A, B, A, C$. Then, from AS, they identify the maximum probable activity using the following equation:

$$P(t|c) = \frac{N_{ct} + 1}{\sum_{t' \in V}(N_{ct'} + 1)}$$

where N_{ct} is the number of times a particular sensor id occurs in activity 'c'. V is the set of sensor ids.

After recognizing the started activity, a Sensors Activity Pattern Matching (SAPM) technique is used to compare it to the detected activated sensors in order to detect any irregularities. The Two wellness functions β_1 and β_2 are also used to check its duration time.

$$\beta_1 = 1 - \frac{t}{T}$$

$$\beta_2 = 1 + \left(1 - \frac{T_a}{T_n}\right)$$

where β_1 is the wellness function of the elderly based on the measurement of inactive duration of appliances, t is the time of inactive duration of all appliances (i.e.) duration time no appliances are used, T is the maximum inactive duration during which no appliances are used, leading to an unusual situation, β_2 is the wellness function of the elderly based on excess usage measurement of appliance, T_a is the actual usage duration of appliance and T_n is the predicted duration use of appliance for the day of the week. A time series forecasting technique is used to take into account trend and seasonality effect in T_n prediction.

Results presented in this work were satisfactory but using a supervised manner to create regular behaviors may cause a problem dealing with new activities or modified ones. Activities sequence creation can also be problematic since, in real time, for each activated sensor, its belonging is checked for all activities.

Several approaches have proposed an unsupervised manner for regular behaviors creation. A simplistic method was presented by Suryadevara et al. (2011), where the status of each sensor, active or inactive, and its duration time are recorded. To differentiate week day from week end behaviors, for each sensor two sequences of its use are created. A sequential pattern matching process is then applied for detecting irregularities.

This simplistic method has several disadvantages including its inability to handle even small changes. For example, if the home occupant wake up half an hour later, or earlier, all the day's activities will be classified as irregular.

Jurek et al. (2014) have also proposed an approach with unsupervised manner for regular behaviors creation based on data mining techniques (Kautkar 2014). They use the k-mean (Garg and Malik 2014) algorithm on a data warehouse recorded from stream of sensor activations in order to create homogenous clusters that represent activities. For activity recognition, they used a classifier to designate the corresponding cluster for detected activated sensors based on their distances from each centroid of clusters. As known, for using k-means, the number P_k of clusters to be created must be specified in advance. In this work, they only assume that more empty clusters will be obtained using high values of P_k and they don't use any criterion function for choosing the optimal one. Another problem with this approach is the absence of any temporal information as duration time of an activity or between two sensors that will help in irregularities detection.

Several other approaches have been proposed using various techniques. Our approach, which will be detailed in the next section, resembles to Suryadevara et al. (2013) with some differences. We propose an unsupervised algorithm for regular behaviors creation represented by activities models that contain ordered activated sensors with duration time between each sensors couple. We also use time series forecasting technique, not for activity duration time prediction, but for predicting activities started times that will help us reducing the number of activities models while recognizing the started activity. For started activity search we opted for a Bayesian network.

3 Proposed activity recognition approach

The three stages of assistance are schematized in Fig. 2. The first one is an observation stage where all sensors measurements are recorded as shown in Table 1. The observation day's number and the very frequent sensors measurements make the data warehouse huge and unusable. Thus, reduction of the data warehouse dimensionality, without losing relevant information, is required. Instead of recording all sensors values at each measurements, we record only activated sensors with their activation time. We consider that a sensor has been activated when its stats changes (Boolean sensor) or when its numerical value greatly changes (small changes are considered as noise). From Table 1, by comparing successive rows we obtain a new data warehouse schematized in Table 2.

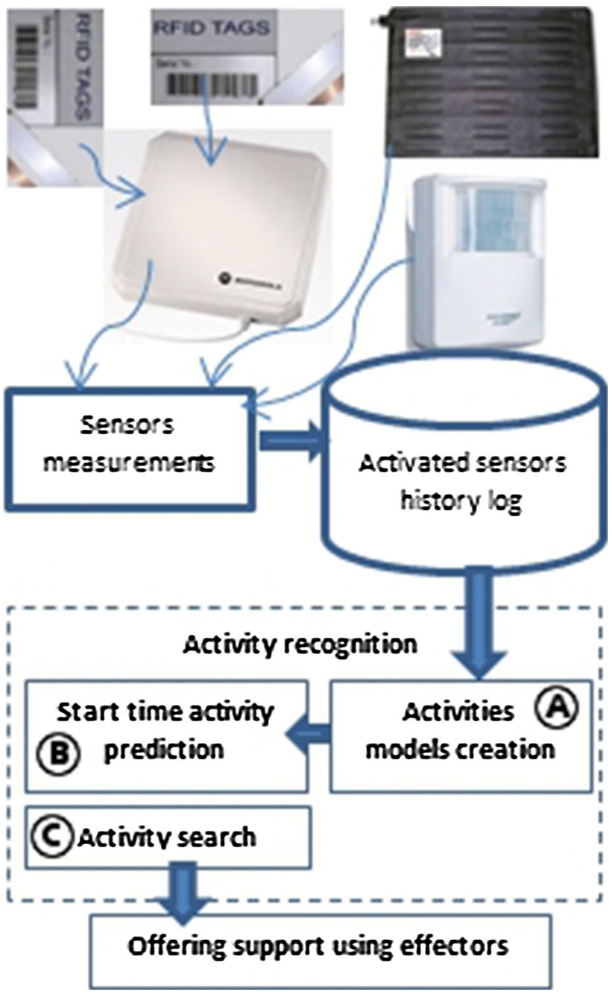

Fig. 2 Technological assistance stages

Table 2 Activated sensors sequences

Date	Activated sensors
01/03/2015	$Sensor_1(Time_1)$, $S_7(T_2)$, $S_2(T_3)$, ...
02/03/2015	Sensor 4($Time_1$), ...

As already mentioned, the second stage is the most challenging one in this process. It allows recognizing started activity in order to be able to detect any irregularities that will requires the use of the third stage effectors. Our activity recognition stage takes as input the data warehouse created in observation stage, Table 2, and latest activated sensors, and outputs the activity being performed. We separate this stage on three steps; activities models creation that finds all closed frequent patterns in Table 2, activities start time prediction that will be used to reduce the number of activities considered in the third step which is activity search.

3.1 Activities models creation

The activation of one or more sensors reflects the interaction of the smart home occupant with its environment. In other words, activated sensors reflect some occupant actions. This means that an activity is a particular sequence of sensors. Since we are interested in ADL, these activities will be frequent and activity detection will result in a search for closed frequent pattern (Agrawal and Srikant 1995). To detect new activities or changes in existing ones, we need an unsupervised algorithm that will be executed every month. Not need to be a real time algorithm. It can be run at night when the observed person is asleep.

Because of two specificities of our case, we chose to create our own activities models creation algorithm. Firstly, We wanted a total order between sensors in the activity model so the ambient agent will be able to know and propose the next sensor if needed. We are aware that no one will repeat an activity in the same way all the time, but it is also true that we all have our habits and lifestyle that will ensure that a large percentage of a repeated activity will be the same. This percentage, the frequency threshold, plays a key role in our algorithm. It will be discussed in the validation section. Secondly, activity model must contain duration time between two adjacent sensors to allow the ambient agent to detect irregularities. This duration should be as accurate as possible because it can differentiate activities. For example, if the duration time of *boiling water* is around two minutes, the activity may be *making tea*, but if its duration time is around fifteen minutes, the activity may be *making pasta*. For this reason, we have introduced the notion of homogeneity explained in the next subsection.

3.1.1 *Problem definition*

From an events sequences set S, like Table 2, we have to find frequent closed homogeneous subsequences. Given a set E of sensors, an event is a pair (A, t) where $A \in E$ is a sensor and t (an integer in seconds) is the sensor activation time. For instance, Fig. 3 describes the following two events sequences set where the symbol x depict the smart cutting explained later:

$s_1 = <(A, 01), (B, 04), (C, 06), (D, 28), (A, 36), (A, 41), (B, 56), (E, 58), (A, 59)>$

$s_2 = <(E, 08), (A, 11), (B, 30), (E, 32), (A, 42), (B, 46), (C, 49)>$

Definition 1 (*Occurrence*) A subsequence $s' = <(A'_1, t'_1), (A'_2, t'_2), \ldots, (A'_k, t'_k)>$ occurs in an events sequence $s = <(A_1, t_1), (A_2, t_2), \ldots, (A_n, t_n)>$ if there exists at least a subsequence $s'' = <(A''_1, t''_1), (A''_2, t''_2), \ldots, (A''_k, t''_k)>$ of s such that $A'_1 = A''_1, A'_2 = A''_2, \ldots, A'_k = A''_k$.

Because an activity can be performed at any time and hardly with the same delays between sensors, we did not specify any special conditions on time in this definition, what will be done by introducing homogeneity notion.

Definition 2 (*Frequent subsequences*) An events subsequence is frequent if its number of occurrence in an events sequences set is greater than or equal to a user predefined threshold. It should be noted that a subsequence may occur several times in the same sequence such as an activity may be performed many times in the same day.

From Fig. 3, if the threshold is two, the subsequence $s'_1 = <(A, t_1), (B, t_2), (C, t_3)>$ is frequent because its frequency is equal to the threshold.

Definition 3 (*Closed frequent subsequences*) A frequent subsequence is closed if it remains frequent after reducing its frequency by its number of occurrences in any other frequent closed subsequence. The notion of closeness is used to avoid considering every part of an activity as a separate activity except when it's the case. We can think of making coffee as a part of making breakfast or as an individual activity.

If we come back to our example, $s''_1 = <(A, t_1), (B, t_2), (E, t_3)>$ is a frequent closed subsequence because it occurs 2 times in S and doesn't occur in any other frequent closed subsequence. But $<(A, t_1), (B, t_2)>$ is not frequent because its frequency is equal to zero; it occurs 4 times in S but occurs 2 times in s'_1 and 2 times in s''_1.

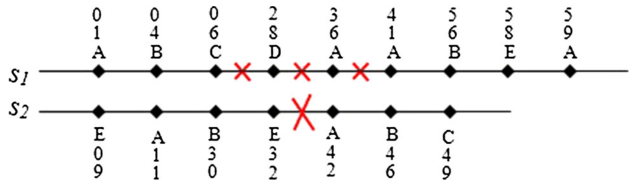

Fig. 3 Two events sequences set showing first cuts when frequency threshold is equal to 2

Proposition 1 (Homogenous interval) *Let* $<(e_1, t_1), (e_2, t_2)>$ *be a two adjacent events , and let* I *be the ordered set of time differences* $(dt_{1,2} = t_2 - t_1)$ *of all their occurrences in S* $dt_{1,2}, dt_{2,3}, \ldots, dt_{n-1,n}$, *I is an homogenous interval if and only if all its elements are close enough to its median. I is represented by its ordered events and its minimal and maximal values*: $I_{e1,e2}[t_{min}, t_{max}]$.

Homogeneous intervals guarantee that their representation will not be affected by an error or an exception. If usual duration between two adjacent sensors is between 5 and 15 s, and once, because of an error, it was 200 s, the homogeneous interval will be represented by [5–15] instead of [5–200]. Our algorithm uses the fuzzy C-means [25] to transform I to one or more homogeneous intervals.

Definition 4 (*Sensors couple*) A sensors couple is a two adjacent sensors with one frequent homogenous interval. Two different sensors couple may have the same sensors but different frequent homogenous intervals. As shown in Fig. 3, there are two different sensors couple with the same sensors A and B. The first one has a frequent homogenous interval [3–4] and the second one [15–19].

Proposition 2 (Homogenous subsequences) *A subsequence* $<(e_1, t_1), \ldots, (e_i, t_i), (e_j, t_j), \ldots, (e_n, t_n)>$ *is homogenous if every time difference between any two adjacent events,* e_i *and* e_j, *belongs to the interval of a sensors couple with the same sensors*: $(dt_{i,j} = t_j - t_i) \in I'_{e_i,e_j}[t_{min}, t_{max}]$.

To demonstrate the importance of the concept of homogeneity, if we go back to Fig. 3, if sensor B is activated after 3 s from the activation of A, the ambient agent can predict that the sensor C will be activated in two to 3 s. But, if sensor B is activated after 17 s from the activation of A, the ambient agent can predict that the sensor E will be activated in 2 s.

Proposition 3 (Activity) *An events sequence* $a = <(a_1, t_1), (a_2, t_2), \ldots, (a_k, t_k)>$ *is an activity if and only if a is a closed frequent homogeneous subsequence in S.*

Definition 5 (*Activity model*) An activity model is defined by:

$am = <Am_1, Am_2(I_1), Am_3(I_2), \ldots, Am_{k'}(I_{k'-1})>$ such that:

Am_i a sensor $\in E$ for all $i = 1, \ldots, k'$;

$I_j = [t_s - t_e]$ is the frequent homogenous interval of the sensors couple Am_{j-1} and Am_j for all $j = 2, \ldots, k'$.

An activity a matches an activity model am if:

$k = k'$;

$a_i = Am_i$ for all $i = 1, \ldots, k$;

$(t_i \in t_{i-1}) \in I_i$ for all $i = 2, \ldots, k$.

3.1.2 Activity pattern mining

Activity pattern mining, as defined in this paper, takes a temporal sequences database and a frequency threshold as input and discovers patterns that are totally ordered on events. Existing algorithms create special structures such as trees, to reduce the task complexity or combine frequent patterns in order to find longer ones. Unlike them, our algorithm works in the opposite way by removing progressively infrequent subsequences.

The proposed algorithm consists of two parts. The first one, detailed in Algorithm 1 (T2NTDB), aims to transform the input temporal database into a non-temporal database with the possibility of recovering all relevant temporal data. This is a very interesting point, because it reduces the complexity of dealing with temporal data for algorithms that already take temporal database as input and allows those that weren't conceived to deal with temporal data to work on temporal databases.

A first pass over Table 2 in T2NTDB allows the identification of all frequent adjacent sensors and their time differences. Table 3 shows the result of this first pass if we apply it on the events sequences set showed in Fig. 3 with a threshold equal to 2. Because intervals created from time differences may not be homogenous, CHI Algorithm, explained later, is called to transform Table 3 to Table 4 which contains only frequent sensors couples.

After that, T2NTDB makes a second pass over the whole sensor log in order to create the new event sequences set S' by cutting between adjacent sensors that don't belong to T' and substituting the others with their index in T'. By belonging to T' we mean that a line k of T' exists such as the first and second sensor in k are equal to the first and second adjacent sensors and the adjacent sensors time difference belongs to the same line interval.

As shown in Fig. 3, the adjacent sensors $(E, 58), (A, 59)$ of s_1 is not cut because of the last line of T'(Table 4); $59 - 58 = 1 \in$ [1–2], and will be substituted by 4, the index of the last line in T', in S'. While the adjacent sensors $(E, 32), (A, 42)$ is cut because it doesn't belong to T'; $42 - 32 = 10 \notin$ [1–2].

Table 3 Time differences of frequent adjacent sensors

Sensor$_1$	*Sensor*$_2$	Frequency	Time differences
A	B	4	3, 15, 19, 4
B	C	2	2, 3
B	E	2	2, 2
E	A	3	1, 2, 10

Table 4 Frequent sensors couples

Sensor$_1$	*Sensor*$_2$	Frequency	Frequent homogenous interval
A	B	2	[3–4]
A	B	2	[15–19]
B	C	2	[1–3]
B	E	2	[2–2]
E	A	2	[1–2]

For our example, the new event sequences set S' will be:
$< 0, 2 >$
$< 1, 3, 4 >$
$< 4, 1, 3 >$
$< 0, 2 >$

The subsequences number in the new set might be greater than the original one, but they are certainly shorter which reduces the complexity of mining frequent patterns. Index used in substitution allow, at any time, recovering relevant temporal information omitted in the new set.

Algorithm 1 T2NTDB: Transforming temporal subsequences set to a non-temporal one

```
Input: a temporal sequences set $S$, Threshold $F$
Output: a non-temporal sequence set $S'$, an array $T'$ of sensors couples
1: For each sequence $s \in S$ do
2:     For each adjacent sensors $c \in s$ do
3:         if $c \in T$ then                ▷ $T$: array shown in Table 3
4:             $c.frequency++$
5:         else
6:             ADD $c$ to $T$
7:             $c.frequency \leftarrow 1$
8:         end if
9:     end for
10: end for
11: For each $c \in T$ do
12:     if $c.frequency < F$ then
13:         Delete $c$
14:     end if
15: end for
16: Call CHI Algorithm               ▷ To create $T'$ (Table 4)
17: For each sequence $s \in S$ do
18:     Create an empty new sequence $s'$
19:     For each adjacent sensors $c \in s$ do
20:         if $c \in T'$ then
21:             ADD its index ($k$) in $T'$ to $s'$
22:         else
23:             if $s'$ is not empty then
24:                 ADD $s'$ to $S'$
25:                 Clear $s'$
26:             end if
27:         end if
28:     end for
29: end for
30: return $S'$ and $T'$
```

CHI Algorithm transforms frequent adjacent sensors with their time differences to one or more sensors couples.

We chose to use the Fuzzy C-Means (FCM) (Bezdek and Ehrlich 1984) clustering algorithm because it allows a time value to belong to more than one cluster. This characteristic is useful in our approach because it can increase the size of certain time intervals. To determine the optimal number of clusters, we execute it multiple times on each adjacent sensors using an increasing number each time. The criterion we use to choose the best number of clusters is C_N; the average deviation of each value from the median (Md) of its most probable cluster. When more than one cluster is analyzed, the criterion value is the sum of each cluster average. The algorithm considers that the optimal clusters number is N if the $(N+1)$ clusters criterion value doesn't improve significantly the one with N clusters.

$$C_N = \sum_{i=1}^{N} \left[\frac{\sum_{j=1}^{ValueNum} |x_{ij} - Md_{ij}|}{ValueNum} \right] \quad (1)$$

Algorithm 2 CHI: Creating sensors couples

```
Input: an array of adjacent sensors with their time differences $T$, Threshold $F$
Output: : an array of frequent sensors couples $T'$
 1: $N \leftarrow 1$
 2: For each $c \in T$ do
 3:     Calculate the criterion value $C_1$ (for $N = 1$ cluster)
 4:     $N++$
 5:     Apply FCM on time differences using $N$ clusters
 6:     Calculate the new criterion value $C_N$
 7:     if $C_{N-1} - C_N > \varepsilon$ then ▷ $\varepsilon$ is experimentally defined
 8:         go to 4
 9:     else
10:         For each Clucter $cl$ in the division $N-1$ do
11:             if the number of cluster elements $n \geq F$ then
12:                 ADD $c$ to $T'$ ($n$ as frequency, $cl$ as cluster)
13:             end if
14:         end for
15:     end if
16: end for
17: return $T'$
```

Returning to our example, the list of all time differences between couple (A, B) of Fig. 1 is 3, 4, 15, 19. In this case, the median for one cluster is 9.5 and C_1 is 6.75. For N equal to two, FCM finds two clusters, 3, 4 and 15, 19. The median of the first cluster is 3.5 and its average deviation is 0.5. For the second cluster, the median is 17 and its average deviation is 2. In total, the two cluster division criterion value (C_2) is 2.5 which is significantly lower than the value for one cluster. Using three clusters, the total value (C_3) is 2 which doesn't significantly improve (C_2). The optimal number of clusters is then two.

Even if division with $(N+1)$ clusters slightly improves C_N, we choose division with (N) clusters because we need clusters with maximum elements in order to be frequent and could be added to T'. Table 4 shows the content of T' for our example.

It should be noted that couples of array T' are frequent homogenous subsequences. It only remains to find the ones that are closed to find activities models of size two.

The second part of our algorithm is very similar to the first one except the inexistence of temporal data. The details of this part, shown in Algorithm 3, can be summarized as: creating frequent adjacent sensors array then using its indexes for substituting or cutting adjacent sensors. This process is repeated until no sequence remains in S'.

Algorithm 3 Creating activities models

```
Input: a non-temporal sequences set $S'$, Threshold $F$
Output: : activities models $AM$
 1: while $S'$ is not empty do
 2:     Create a new frequent sensors couples array $T'$ from $S'$
 3:     Use $T'$ to create, by substitution, a new sequence set that will be the new $S'$
 4:     Save $T'$ in an array of arrays $TT$
 5: end while
 6: ADD all subsequences of the array $TT$ to $AM$
 7: For each subsequence $s$ of an array $TT[i]$, $i$ from $TT.size()-2$ to 0 do
 8:     For each subsequence $s'$ of an array $TT[j]$, $j$ from $i+1$ to $TT.size()-1$ do
 9:         if $s$ occurs in $s'$ then
10:             $s.frequency \leftarrow s.frequency$ - $s'.frequency$
11:         end if
12:     end for
13:     if $s.frequency > F$ then
14:         ADD $s$ to $AM$
15:     end if
16: end for
17: return $AM$
```

Then, the array of arrays TT, which records all created arrays, is used to recover activities models. Recursive method can be used, but its execution time is much longer than the iterative one that we use. As we duplicated internal

Table 5 Array of last frequent sensors couples

$Sensor_1$	$Sensor_2$	Frequency	Subsequence
0	2	2	$A, B(3-4), C(1-3)$
1	3	2	$A, B(15-19), E(2-2)$

Table 6 Created time series

Activities	Day 1	....	Day N
Wake up	1230	...	0
...	...	...	...
Taking medication	6767	...	6760

events on substitution, redundancy must be removed while recovering them. Table 5 shows the last frequent adjacent sensors array created for our example.

The final step of our algorithm consists of finding closed subsequences that are activities models. All subsequences of the last created array are closed because no higher size subsequences exist. For any subsequence of other arrays, it must be compared to all subsequences of arrays created after its array and its frequency must be reduced by the frequency of any subsequence that contains it. If its frequency is still higher than the threshold, this subsequence is closed and must be added to model activities.

We have chosen this activity model structure in order to include very relevant information that will be used in the assistance process. The perfect order of sensors that compose an activity model allows the ambient agent to know unambiguously, once an activity is recognized, the next sensor to be activated. Moreover, exceeding the time interval for activating a sensor will help the ambient agent inferring that the home occupant has problems finishing the started activity and possibly offers assistance by indicating the next sensor in the activity model. Other information, such as activities start times, will also be used in the next step of activity search.

3.2 Activity start time prediction

In addition to reduce activities models considered in activity search process, activity start time prediction addresses two other problems: considering activity with the closest predicted start time to current time as the recognized activity answers the equiprobability problem that occurs when activated sensors belongs to more than an activity, and the initiation problem that occurs when the home occupant is unable to start the activity and thus no sensor is activated. Activity start time prediction is not a real time Algorithm and can also be executed every night to predict activities start time of the next day.

We chose to implement an activity start time prediction step instead of directly use the activity start time interval median, because an activity start time may change depending on the day of the week. For instance, if the home occupant prepares coffee around 0700 hours each day of the week except Sunday, the activity start time interval median will be 0700 hours and the activity prepare coffee will be considered in activity research executed around 0700 hours for all days including Sunday.

After a thorough study, we found that time series forecasting techniques offer the best solution for predicting activities start times depending on the day of the week using seasonality. A time series is a sequence of observations taken sequentially in time [10] denoted $(X_t)_{t\in\theta}$ where the set θ is the space time. In our case, each activity is a time series, observation days are the space time and activity start times are the observations. In order to create time series, shown in Table 6, a last pass over data warehouse is needed to record activities start times of all days.

In Table 6, a non-zero integer represents the start time in seconds, while 0 indicates that the activity was not done that day.

Time series techniques are based on the notion of autocorrelation to predict future values. This means that successive series values must depend on each other otherwise the series is random and its future values are not predictable. Activities time series are not random because every person has a lifestyle and habits that make the majority of his activities carried out almost at the same time as the previous day or as the same day of the week. In addition, the prediction will be more accurate for the activities that we are more interested to predict like taking medication whose start time is rigorously respected.

Taking into account several parameters such as stationary, autocorrelation, and covariance, we opted for autoregressive integrated moving average (ARIMA) (Joseph 1984) model which is a discrete time linear equations with noise, of the form:

$$\left(1-\sum_{k=1}^{p}\alpha_k L^k\right)(1-L)^d X_t=\left(1+\sum_{k=1}^{q}\beta_k L^k\right)\epsilon_t$$

where p is the autoregressive model order, q is the moving average model order, d is the differentiation order, α_k and β_k are model parameters, ϵ_t is white noise and L the time lag operator: $LX_t = X_{t-1}$.

Predicting future values with ARIMA(p, d, q) is done by finding the three parameters p, d and q that yield the best results. The first parameter, d, is equal to 0 if Xt is stationary otherwise d is incremented until $(1-L)^d X_t$ is stationary. To decide if a time series, X_t, is stationary, KPSS tests (Kwiatkowski et al. 1992) is used. In the KPSS model, time series is represented as a sum of three components:

$$\begin{aligned}X_t &= \xi t + r_t + \epsilon_t\\ r_t &= r_{t-1} + u_t\end{aligned}$$

where t is a deterministic trend, r_t is a random walk process, ϵ_t is a stationary error terms and u_t is an error term with a constant variation σ_u^2.

Then, the following three rules are used to decide if X_t is stationary:

- If $\xi = 0$ then X_t is stationary around r_0;
- If $\xi \neq 0$ then X_t is stationary around a linear trend;
- If $\sigma_u^2 > 0$ then X_t is non-stationary.

The second step of ARIMA model is to find the parameters p and q that better fit the model. For this purpose, the chosen p and q parameters are the ones that minimize the Akaike's information corrected criterion (AICC) (Khim and Shitan 2002). AICC statistics is given by:

$$AICC = -2lnLikelihood(\widehat{\Phi}, \widehat{\theta}, \widehat{\sigma}^2) + [2n(p+q+1)]/[n-(p+q)-2]$$

where $\widehat{\Phi}$ is a class of autoregressive parameters, $\widehat{\theta}$ is a class of moving average parameters, $\widehat{\sigma}^2$ is the variance of white noise, n is the observations number, p is the autoregressive component order, q is the moving average component order.

So far, predictions are only based on previous days observations. For more accurate predictions, time difference between predicted and actual start times must be considered. For instance, if *wake up* predicted start time were 0700 hours, but the home occupant woke up at 0730 hours, predicted *take breakfast* start time must be updated. Thus, when a big difference between an activity predicted start time and its actual start time is detected a second prediction is performed. This time, a time series $(Y_t)_{t\in\theta}$ is created for each activity (X'_t) that takes place around the last activity detected (X_t) where θ is the observation days set and Y_i is start times difference between the two activities at the day i; $X'_i - X_i$. ARIMA is again used to predict time difference between the two activities at horizon $h(Y_h)$. The new predicted start time Y'_h is calculated based on X_t actual start time and the time difference predicted at horizon h:

$$Y'_h = X_h + Y_h$$

3.3 Activity search

Now that activities start times are predicted, we can use Algorithm 4 in order to select activities that will be considered in Bayesian network, calculated and updated their initial probabilities. The most likely activity will be those with the highest probabilities.

Algorithm 4 starts by creating closest activities set based on predicted start times and current time and assigns to each activity in the set an initial probability based on its daily confidence. Then, probabilities are updated to make the closest ones to current time more probable. When a big difference between predicted start time and actual start time of the activity detected is observed, predicted activities start times are recalculated. Probabilities are updated again based on the new predicted start times.

For activity search, a similar Bayesian network is used but this time it takes as input the selected activities by Algorithm 4 and uses their probabilities as initial probabilities. Their probabilities are then updated each time a sensor is activated depending on its belonging to the activity.

Algorithm 4 Bayesian network for activity prediction

```
Input: : All activities models $A$
Output: : Selected activities with their probabilities
1: For each activity $a \in A$ do
2:     $a.PST$ = $a$ predicted start time
3:     if ($|a.PST -$ Current time $Ct| < \epsilon$) then
4:         Add $a$ to $SA$
5:         Probability a.P = (daily confidence)$_{normalized}$
6:     end if
7: end for
8: For each $a \in SA$ do
9:     $a.P = Pa * (\frac{1}{|a.PST - Ct|})_{normalized}$
10: end for
11: if (an activity $a' \in SA$ is detected) then
12:     remove $a'$ from $SA$
13:     if ($|a'.PST - a'.DST| > \epsilon'$) then
14:         For each $a \in SA$ do
15:             create $Y$ (Time series of start times differences)
16:             Predicte $PY$ (time between $a$ and $a'$)
17:             new $a.PST = a'.DST \pm PY$
18:         end for
19:         For each $a \in SA$ do
20:             $a.P = Pa * (\frac{1}{|a.PST - Ct|})_{normalized}$
21:         end for
22:     end if
23: end if
24: Sleep( S seconds)
25: Go to 11
26: return $AM$
```

4 Validation

Before the validation of our approach with real data recorded in a smart home, we used synthetic data to evaluate activities models creation which, we think, can be used in other cases study. In order to generate significant synthetic data, we must know the factors that most influence its performances. That's why we evaluated its execution time, T_e, in the worst case. The worst case happens when sequences are composed of the same event that occurs periodically such that each array, created by the algorithm, will contain one couple. If N is the longest sequence events number, n is the sequences number in the event sequences set and F is the frequency threshold, the execution time T_e can be expressed as:

$$Te = 2n\frac{N(N+1) - F(F+1)}{2} \tag{2}$$

To create all event couples arrays, we pass N times over n sequences of length N, so $Te = nN^2$. The same time is needed for substitution process, $Te = 2nN^2$. To be precise, after each substitution N decreases by one, so it's not $N \cdot N$

but $1 + 2 + \cdots + N = N\frac{N+1}{2}$. To be more precise, the algorithm stops when no frequent sequence remains. That means that it stops when N becomes lower than F, so it's not $1 + 2 + \cdots + N$ but:

$$F + F + 1 + F + 2 + \cdots + N = N\frac{N+1}{2} - F\frac{F+1}{2} \quad (3)$$

According to the above formula, the threshold F is the major factor that influences the algorithm execution time. When F tends to N, Te tends to zero, and when F tends to zero, Te tends to nN^2.

In all cases, the whole log will be read v times, where v is the length of the longest frequent closed homogenous subsequence. The execution time will differ depending on the average with which the whole log will decrease after each substitution.

So, to validate the proposed algorithm, we decided to generate moderately frequent event sequences sets of different sizes. Then, we observed the execution time variation depending on the set size and on the threshold. Figures 4 and 5 show obtained results.

As shown in the graph of Fig. 4, until 20,000 events, increasing set size increases almost linearly the execution time. After that, execution time increases considerably to attain 3 h. The same observation can be done on the graph of Fig. 5. Execution time increases almost linearly while the threshold decreases until 8. After that it dramatically increases.

Those observations confirm the finding of worst case analysis; execution time depends more on the whole log average decrease after each substitution.

The question that must be answered now is: what is the expected execution time in our case of ADLs detection. To answer this question, we created a real events sequences set by observing a person in the LIARA laboratory and recording all activated sensors during the first hour after his wake up for a period of 4 weeks.

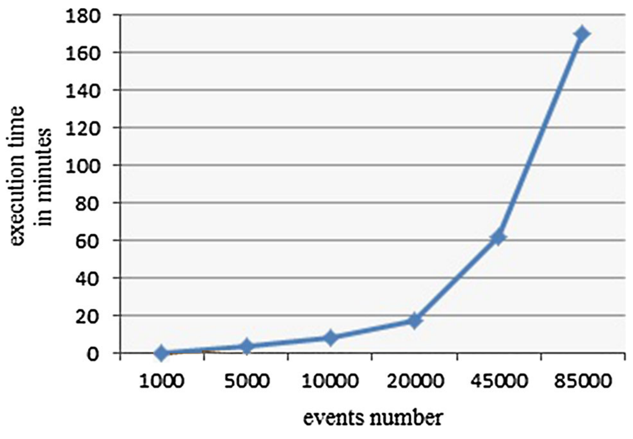

Fig. 4 Execution time depending on set size

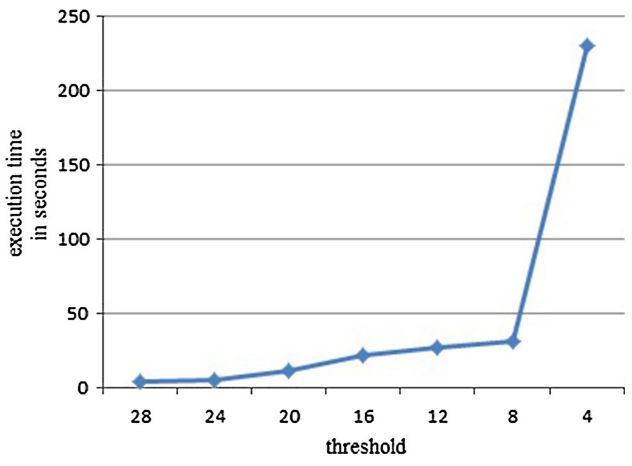

Fig. 5 Execution time depending on threshold

Fig. 6 LIARA

The Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activits (LIARA) possesses a new cutting edge smart home infrastructure that is about 100 m^2 and possesses around a hundred of different sensors and effectors. Among the sensors, there are infrared sensors, pressure mats, electromagnetic contacts, various temperature sensors, light sensors and eight RFID antennas. The Fig. 6 shows a cluster of images from different parts and angles of the smart home.

All LIARA sensors send their measurements every 500 ms. They are classified in two categories; Boolean and numerical. Only activated sensors are recorded in the database using the structure in Table 2. We consider that a sensor is activated when its state changes (Boolean) or when its value greatly changes (numerical). A small change in value is considered to be noise and is therefore ignored. Some other restrictions are applied before adding any activated sensor to the database. For example, if an RFID or a motion detector sensor is activated multiple successive

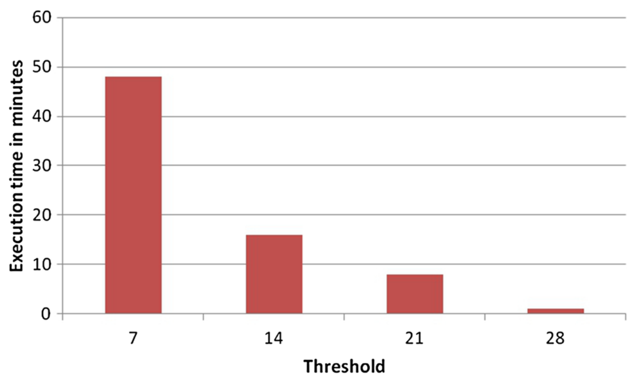

Fig. 7 Execution time depending on threshold

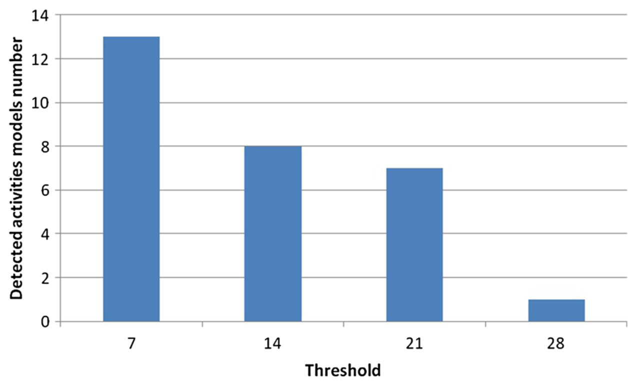

Fig. 8 Detected activities models number depending on threshold

times, it's recorded only once. Otherwise it will be impossible to detect this activity because the activation number cannot be precise. In addition, while assisting the occupant, we want to show him the next object to be used, not how to use it.

The obtained database is composed of around 1100 events. Six activities were observed:

1. Wake up
2. Use toilet
3. Wash hands
4. Take shower
5. Prepare coffee
6. Leave house

We tried different thresholds to evaluate time execution and activities models detected. As we observed the occupant for 28 days, thresholds were chosen between 28 and 7 to detect activity performed at least once in 4 days. Figures 7 and 8 show the obtained results.

High threshold (28) decreases the execution time but doesn't allow the detection of all activities models. That can be explained by the fact that some activities are not performed every day and sometimes, as already mentioned, some activities are performed differently. At the opposite,

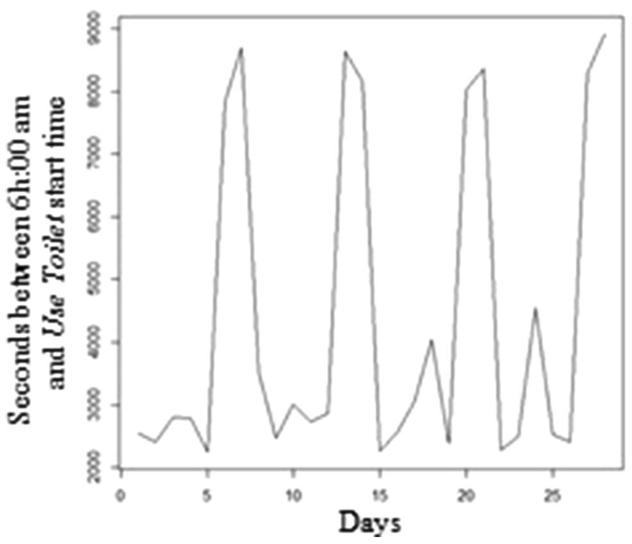

Fig. 9 Four weeks plot

low threshold (7 and 14) increases the execution time but some closed homogenous subsequences become easily frequent and are considered as activities. According to Figs. 7 and 8, the optimal threshold is around 75.

In order to define a person habits, she must be observed at least for 2 months. If we suppose that she is active 10 h a day at home, we can predict that events number for the 2 months will be around 70,000. We can also predict, according to the last figures, that the execution time will be approximately two and a half hours. For this reason we plan to execute activities models detection each 2 months at night while the home occupant is sleeping. This way, the execution time will not be a problem and the activities models, used in the real time assistance, will consider any occupant habits change.

A last pass over the data warehouse allowed us to create six time series, one for each activity. We used the first 3 weeks for forecasting and used the last week to validate the results. 83 % of detected activities were well predicted. Only Leave house seems to be random. For instance, Fig. 9 shows the 4 weeks plot of Use toilet time series and Fig. 10 shows 3 weeks plot and the next predicted week.

The chosen model for this time series was ARIMA(1,0,0) which means that the series is stationary ($d = 0$) and the minimal value of AICC (358.67) was found with $p = 1$ and $q = 0$. As shown in the two lasts figures, predicted values are close enough to actual ones except for the 24th day; predicted value is 3101.941 while the actual value is 4559. By observing Wake up predictions, which precede Use toilet, a big difference between predicted start time (3855) and actual start time (4546) for the 24th day is detected. According to Algorithm 4, a new time series Y_t is created by differentiating Wake up and Use toilet start times and Use toilet predicted

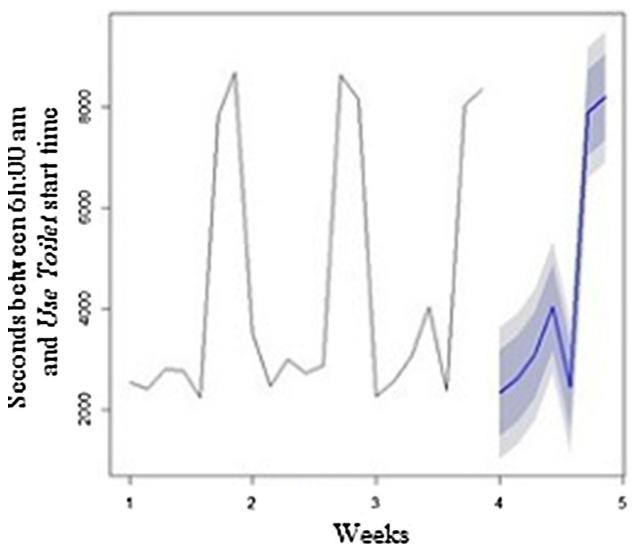

Fig. 10 Use toilet 3 weeks plot and predicted week

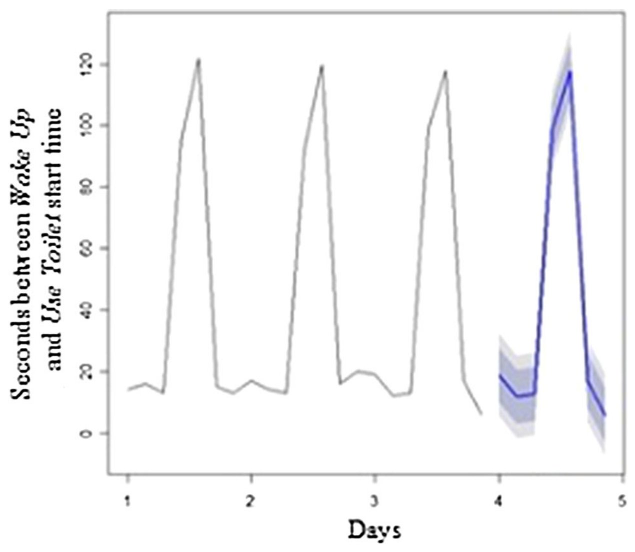

Fig. 11 Three weeks plot and predicted week

start time is recalculated. Figure 11 shows the predicted week for this time series.

The new Use toilet predicted start time is equal to Wake up start time (4546) plus Y24 (12.714) which is 4558.71 $\approx$ 4559 (its actual start time).

A general evaluation of our approach is made by calculating the percentage of performed activities having the maximum likelihood calculated by Algorithm 4 during the fourth week. Figure 12 shows the obtained results.

From Fig. 12, we can see that Wake up and Wash hands were perfectly performed as predicted. The Wake up result was quite expected because it is always the first performed activity. The same observation can be made for Wash

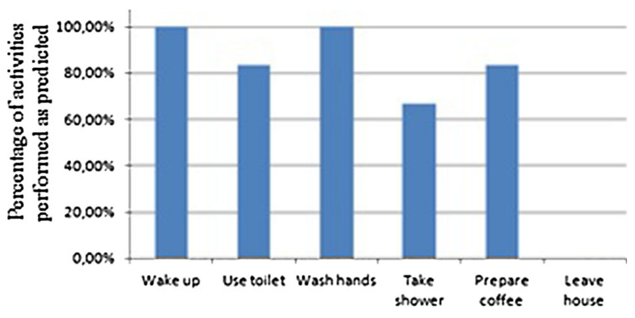

Fig. 12 Three weeks plot and predicted week

hands which always follows Use toilet. The result of Take shower wasn't optimal because this activity was not carried out regularly. The worst result was for Leave house which we couldn't forecast. What can be deduced from Fig.12 is best results were for activities performed less randomly, which is true for most activities.

5 Conclusion

In this article, we presented a comprehensive approach activity recognition in smart homes. First, we used smart home sensors to observe environment changes which are considered as home occupant actions. Then, the sensors history log is transformed and analyzed to detect frequent sensors sequences which are activities that the home occupant is used to perform. After that, in order to reduce the number of activities likely to be performed and facilitate activity recognition, time series forecasting techniques are used to predict activities start times. Thereafter, activity recognition is done by modifying activities probabilities depending if they contain detected activated sensors.

Detecting activities models and predicting activities start times results were satisfactory especially for activities performed less randomly. In future work, we will explore the use of multivariate time series in order to ameliorate our prediction system by considering relations between activities.

References

Agrawal R, Srikant R (1995) Mining sequential patterns. In: 11th international conference on data engineering (ICDE95). IEEE Computer Society, Taipei, Taiwan, pp 3–14

Alzheimer Society (2014) Rising tide: the impact of dementia on Canadian Society

Augusto JC, Nugent CD (2006) Smart homes can be smarter. Springer, Berlin

Bezdek JC, Ehrlich RW (1984) FCM: the fuzzy C-means clustering algorithm. Comput Geosci 10:191–203

Box GEP, Jenkins GM, Reinsel GC (1994) Time series analysis, forecasting and control, 3rd edn. Prentice Hall, Englewood Clifs

Chen L, Hoey J, Nugent C, Cook D (2012) Sensor-based activity recognition. IEEE Trans Syst Man Cybern Part C Appl Rev 42:790–808

Dupuis SL, Epp T, Smale B (2004) Caregivers of persons with dementia: roles, experiences. Supports and coping: A literature review, Ontario

Garg T, Malik A (2014) Survey on various enhanced K-means algorithms. Int J Adv Res Comput Commun Eng 3(11)

Jalal A, Jeong TK, Tae-Seong K (2011) Recognition of human home activities via depth silhouettes and transformation for smart homes. In: SHB2011—5th international symposium on sustainable healthy buildings, Seoul, Korea

Joseph BS (1984) Microcomputer based interactive analysis of univariate and multivariate ARIMA models. In: WSC 85 proceedings of the 17th conference on einter simulation

Jurek A, Nugent C, Bi Y, Wu S (2014) Clustering-based ensemble learning for activity recognition in smart homes. Special Issue of MDPI Sensors Journal for UCAm I & IWAAL 14:12285–12304

Kautkar R (2014) A comprehensive survey on data mining. Int J Res Eng Technol 3(8):185–191

Khim L, Shitan M (2002) The performance of AICC as an order selection criterion in ARMA time series models. Pertanika J Sci Technol 10:25–33

Kwiatkowski D, Phillips P, Schmidt P, Shin Y (1992) Testing the null hypothesis of stationarity against the alternative of a unit root. J Econom 54:159–178

Metras H (2005) RFID tags for ambient intelligence: present solutions and future challenges. In: Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies, ACM, New York, pp 43–46

Sadri F (2011) Ambient intelligence: a survey. ACM Comput 43:36–66

Srinivasan R, Chen C, Cook DJ (2010) Activity recognition using Actigraph sensor. In: 4th international workshop on knowledge discovery from sensor data

Suryadevara N, Gaddam A, Rayudu R, Mukhopadhyay S (2011) Wireless sensors network based safe home to care elderly people: behaviour detection. Procedia Eng 25:96–99 (Elsevier)

Suryadevara N, Mukhopadhyay S, Wang R, Rayudu R (2013) Forecasting the behavior of an elderly using wireless sensors data in a smart home. Eng Appl Artif Intell 26:2641–2652 (Elsevier)

Van Tassel M (2011) Guidelines for increasing prompt efficiency in smart homes according to the residents profile and task characteristics. Springer, p 6719