Preference Learning on an OSGi Based Home Gateway

Md. Kamrul Hasan, Kim Anh Pham Ngoc, Young-Koo Lee, Sungyoung Lee

Abstract — *The goal of ubiquitous computing is to create intelligent environment. To make the environment adapt rationally according to the desire of users, the system should be able to guess users' interest, by learning users' preferences. Users' preferences are sometimes conflicting and needs to be resolved. When many users are involved in a ubiquitous environment, the decisions of one user can be affected by the desires of others. This makes learning and prediction of user preferences difficult. In this paper we prove that learning and prediction of user preference is NP-Hard. So, we propose Bayesian RN-Metanetwork, a multilevel Bayesian network to model user preference and priority. This is a semi optimal online learning approach. By using game theory we prove that the method we use will certainly converge after a while. We also provide implementation details of the metanetwork on an OSGi based home gateway*[1].

Index Terms — Preference Learning, OSGi, Home Gateway, Bayesian RN-Metanetwork.

I. INTRODUCTION

Ubiquitous computing is mainly about building systems which are useful to users, which "...weave themselves into the fabric of everyday life until they are indistinguishable from it" [1]. Users tend to move around often, doing new things, visiting new places, changing their minds suddenly, and changing their moods, too. Therefore, a helpful system should react according to the changing context of the user.

Absorbing and developing the key ideas of ubiquitous computing as well as context-awareness, a smart environment (e.g. smart office, smart home) contains a large number of invisible sensors and actuators which enable the system to "think and work" based on its own perception of users' context [2]. Let us consider following scenario of a smart office:

"*In the morning, secretary comes to the office. When she enters the room, the light is turned on and the curtain is opened. The music player plays the secretary's favorite song. After that, professor arrives. Because the priority of professor is higher than that of secretary, the music is changed to professor's favorite. When a student enters the office and sits down on the chair in front of professor's desk, knowing that professor is having a meeting with his students, the music will be switched off...*"

The scenario sounds like it has some intelligence involved. It somehow satisfies the "invisible interface" requirement of a ubiquitous computing system, because the user does not need to use any explicit interface to interact with the system. However, this is just an automatic system triggered by events. System developer creates the rules and therefore thinks about what the system should do in each situation. The smart system built in this way, will make the same decision, even if the user changes his preference. We argue that, a real smart system should have three capabilities. First, a smart system should be able to do inference. Second, a smart system should be able to learn by itself. User and developers can act like teachers, but the knowledge should be improved incrementally. Finally, a smart system should be able to solve some difficult problems, such as the conflict among the users [3]. Let us add some details to the above scenario to reflect this idea:

"*The music playing is stopped. Even though the command is from the remote control on the secretary's desk, it does not totally mean that the secretary herself wants to stop the music. Maybe professor uses that remote control, or maybe the secretary does it because professor tells her to do. Moreover, there are some possible reasons that he or she does not want to listen to music at this time. The data from other sensors should be used to detect the person who wants to stop the music and his/her reason, so that the next time when the same situation happens, the music will be automatically stopped.*"

The preference of user changes over time or based on situation. It makes online learning (or adaptation) a crucial requirement. Again, when there are many users in the smart environment, the action of one user can affect others' choice [4]. It raises the challenges of distinguishing the preference of each user as well as resolving the conflicts among different user preferences. The introduction of probabilistic model can handle these uncertainty and adaptive prioritization of users. As direct feedback from user is still unrealistic to collect from the user, we base the utility function on the user control commands.

In this paper, we discuss Bayesian RN-Metanetowok [5] along with online adaptation for preference learning with conflict resolution scheme. The algorithm has been implemented on an OSGi based gateway to learn to preferences of users in a smart environment.

The rest of the paper is organized as follows: section II describes related works; in section III we prove that multi-user preference learning is NP. Section V-VII describe the

[1] This research was supported by Ministry of Knowledge Economy, Korea, under ITRC program supervised by Institute of Information Technology Advancement (IITA-2009-C1090-0902-0002). This work was also supported by KOSEF grant funded by the Korean government (MEST, No. 2008-1342), and was supported by Basic Science Research Program funded by National Research Foundation (2009-0076798).

Md. Kamrul Hasan is with Ubiquitous Computing Lab, Kyung Hee University, South Korea. (email: kamrul@oslab.khu.ac.kr)

Kim Anh Pham Ngoc is with Ubiquitous Computing Lab, Kyung Hee University, South Korea. (email: anhphamnk@yahoo.com)

Young-Koo Lee is with Department of Computer Engineering, Kyung Hee University, South Korea. (email: yklee@khu.ac.kr)

Sungyoung Lee is with Ubiquitous Computing Lab, Department of Computer Engineering, Kyung Hee University, South Korea. (email: sylee@oslab.khu.ac.kr). Prof. Sungyoung Lee is the Corresponding Author.

Contributed Paper
Manuscript received July 15, 2009

0098 3063/09/$20.00 © 2009 IEEE

Authorized licensed use limited to: KYUNGHEE UNIVERSITY. Downloaded on November 3, 2009 at 20:11 from IEEE Xplore. Restrictions apply.

Metanetwork and its adaptation. Section VIII describes the convergence property. Section IX and X are implementation and results respectively. We conclude in section XI.

II. RELATED WORKS

Reactive Behavioral System (ReBa) [6] was among the first to propose ubiquitous user preference learning in an intelligent workspace. The system mentions user preference activation using rules based on group activity like meeting, etc. ReBa also tries to resolve conflict among users by setting rules. Rules do not scale well and does not meet the changing need of the users.

To resolve conflicts for Context-aware Media services in smart home environments, Shin.et.al.[7] applies Naïve Bayes classifier to resolve conflicts among users. The system sums up the preferences of users who are collided with each other and recommends the specific contents ordered by the summed preference. However, the system needs to collect data and retrain the classifier again to reflect users' change of preferences.

This paper presents an online algorithm for ubiquitous user preference learning and conflict resolution. The basic idea was published in an earlier paper [8]. In this extended version we added theoretical aspects of preference learning, and added experimental results.

Preference learning in Ubiquitous environment is getting more and more focus now-a-days. Yang.et.al [9] proposes a template based (source device, destination device, media transferred, etc.) user activity detection in a UPnP enabled smart environment. Here the activity actually refers to different multimedia services the system can provide, such as reading newspaper, or watching movie. The system does not refer to user's current context like location, high level activity into consideration. Also it is not outlined how the system will learn the multimedia service preferences.

Adaptive Activity-Based Middleware [10] resolves conflicts at service discovery level and provides alternative resources to user, if it is already assigned to other user. However, if there is only one resource, and a high priority user is present, the middleware will not be able to favor the high priority user.

III. MULTI-USER PREFERENCE PREDICTION

In this section we establish that multi-user preference learning is an NP problem.

Theorem 1: *Given a set of* η *users present in the environment, the task is to predict the collective preference of the* η *users over a set of* L *service options, so that the system's predicted preference is not altered. The problem is* NP - Complete .

Theorem 2: *Given a set of* η *users and a set of* L *service options, determining whether all the users' preferences fall into a single service option is* NP - Complete .

Theorem 3: *Given a set of* η *users and a set of* L *service options, determining the minimum number of service options where all the users' preferences fall into, is* NP - Hard.

Proof of Theorem 2 and Theorem 3: We prove the theorems by reducing them to Set-Cover problem [11]. "Given a universe U and a family S of subsets of U, a cover is a subfamily $C \subseteq S$ of sets whose union is U. In the set covering decision problem, the input is a pair (U, S) and an integer k; the question is whether there is a set covering of size k or less. In the set covering optimization problem, the input is a pair (U, S), and the task is to find a set covering which uses the fewest sets" [12]. The decision version of set covering is NP - Complete , and the optimization version of set cover is NP - Hard [11][12].

Let's take the set of service options L as S and η users' preferences as the universe, U. If a user's preference $u_i \in U$ falls in $S_i \in S$, then S_i contains u_i. So, according to set cover decision problem determining a set covering of size 1 $[S_i = \{U\}]$ is NP - Complete . This means that determining whether there is a single service option that is preferred by all the users is NP - Complete . Similarly, finding the least number of service options that is preferred by all the users is NP - Hard .

As it is NP-Complete to determine if there is a single service option preferred by all the users and NP-Hard to find out the least number of service options preferred by all the users, the system has to guess the service option to be initiated. It might happen that the service option initiated by the system is altered. Determining if the service option will be altered is also NP-Complete.

Proof of Theorem 1: We assume only two parties (SU), first, the system (S), and second, all the users (U). We assume L service options. If both the parties fall into the same service option, the system initiated service option will not be altered. However, computationally the result cannot be predicted. Because, finding out a set covering of size 1 for the pair (SU, L) is NP-Complete. So, we propose a semi-optimal solution by using Bayesian RN-Metanetwork.

IV. BAYESIAN RN-METANETWORK

The Bayesian RN-Metanetwork is a set of Bayesian networks, which are put on two levels in such a way that the distribution of probabilistic networks on first level depends on the local probability distributions associated with the nodes of the second level network. The Bayesian RN-Metanetwork is a triplet:

$$\text{RMBN} = (\text{BN0}, \text{BNS}, \text{R}) \quad (1)$$

where BNS = {BNS1, BNS2, ... BNSn} is a set of sets of Bayesian networks in first layer and BN0 is the second level

Authorized licensed use limited to: KYUNGHEE UNIVERSITY. Downloaded on November 3, 2009 at 20:11 from IEEE Xplore. Restrictions apply.

Bayesian networks; R = {R1 ... Rn} is a set of inter level links. The probability distribution of each Bayesian network is included inside it. Each R_i is a link "vertex – network set" meaning that stochastic values of vertex v_i in the network BN0 correspond to the distribution of one set of Bayesian networks in the first level.

We use the Bayesian RN-Metanetwork to learn both user preference and priority at the same time, as well as to resolve the conflict among many user preferences. Our approach is special in the following aspects:

- Very suitable for multi-agent systems
- The adaptation algorithm is fully described
- The priority and preference of each user is calculated separately which helps knowledge reuse.

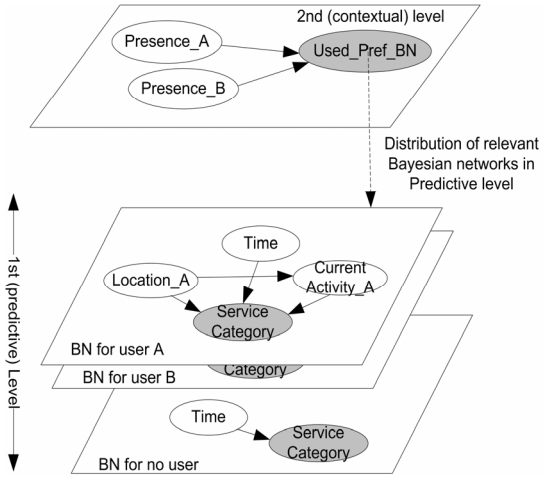

Fig. 1. A Bayesian RN-Metanetwork to learn service preference of 2 users A, B in case of situation-independent priority.

V. Modeling User Preference and Priority using Bayesian RN-Metanetwork

When many users are present in a smart environment and each user has his or her preference about a certain service, the last decision to select the service is related to the priorities of the users. The user priority can be categorized into 2 types: situation-independent priority and situation-dependent priority. Situation-independent priority means that the priority of a user does not change when situation changes. In contrast, situation-dependent priority changes when situation changes. The user preference in case of situation-independent priority can be modeled using the 2-level Bayesian RN-Metanetwork. Figure 1 shows the Bayesian RN-Metanetwork for modeling user preference about ubiquitous services in a 2-user system. We can see that the user priority in this case depends only on the user presence. In case of situation-dependent priority, the 2nd (or Meta) level is replaced by a metanetwork (see Figure 2).

In the model, only two users are assumed for simplicity of equations and explanations. It is straightforward to extend the model for more than two users. It is also possible to add extra causal nodes in the first level.

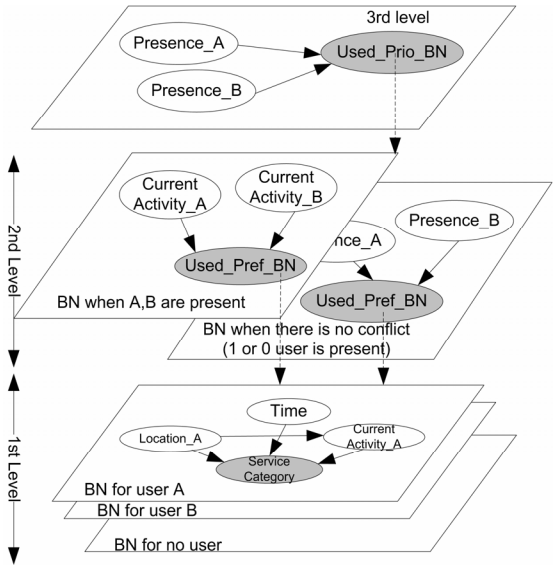

Fig. 2. A Bayesian RN-Metanetwork to learn multimedia service preference of 2 users A, B in case of situation-dependent priority. Priority depends on not only user presence but also user activity.

VI. Probability Propagation and Adaptation for 2-layer Bayesian RN-Metanetwork

In this session, we explain the algorithms for Probability Propagation and Adaptation for the basic 2-layer Bayesian RN-Metanetwork as shown in Figure 1. The Bayesian RN-Metanetwork in Figure 1 has parameters:

- 1st level:
  + This level has only one set of Bayesian networks: 2 Bayesian networks modeling the preferences of 2 users and one Bayesian network for the case of no user.
  + The attributes of the Bayesian networks: Time (T), Location_A/B (Li, i=1, 2 for A and B respectively), Current Activity_A/B (dCAi), Service Category (Sx, x=1, 2, 3 for its duplications in 3 networks) and has the values {SV1 ...SVj}.
  + The prior probabilities: P(T); P(Li); P(CAi), P(Sx).
  + The conditional probability P(Si|T, Li, CAi), i=1, 2 and $P(S_3|T)$.
- 2nd level:
  + The attributes: Presence_A/B (Pr_i, i=1, 2) represents the Presence of user A/B with the values {yes/no}
  + The relevance node: Used_Pref_BN (denoted BN) holds the probability to have each Bayesian model in the predictive level with the values {BN1, ... BN3}.
  + The prior probabilities: P(Pr).
  + The relevance probability: P(BN). The conditional probability P (BN|Pr)

A. Probability Propagation

Given the evidence P (Pr), P(BNx) is calculated as:

$$P(BN_x) = P(BN = BN_x \mid \Pr).P(\Pr) \quad (2)$$

Authorized licensed use limited to: KYUNGHEE UNIVERSITY. Downloaded on November 3, 2009 at 20:11 from IEEE Xplore. Restrictions apply.

Then the probability of the target attribute ServiceCategory can be estimated:

$$P(S) = \sum_{x=1}^{3} P(BN_x).P(S_x) \quad (3)$$

with P(Sx) calculated given the evidence T, Li and CAi

$$P(S_x) = \begin{cases} P(S_x \mid T, L_x, CA_x).P(T).P(L_x).P(CA_x) & x = 1,2 \\ P(S_x \mid T).P(T) & x = 3 \end{cases} \quad (4)$$

In other words, the probability of ServiceCategory preference of each user is calculated separately, and then combined with the weight coefficients which are the distributed probabilities of the Bayesian networks in Predictive level. Finally, the target ServiceCategory SVj with highest P(S=SVj) is selected.

B. Adaptation

Each time the system makes a decision about which service category to be selected, it then wait a time interval, t_w for the response of users. There are 2 cases of user responses:

Approval: If there is no response, then the system assumes that the users are pleased with the decision. In this case, the decision together with the evidences will be considered a single sample and be used to update the Bayesian networks conditional probabilities.

Denial: If one user gives a control command to the system to change the selected service category, it means that the user may not be satisfied with the decision of the system. The user's selection together with the evidences will be considered N samples (N>>1) and be used to update the Bayesian networks conditional probabilities.

The adaptation algorithm is based on two assumptions:

Assumption 1: The contribution of one user preference in the combined preference is equal to his contribution in making a decision.

Assumption 2: Every user has the tendency of selecting the option which has highest probability calculated by his preference model. This option is called the most favorite option. Assumption 2 leads to a definition: *A decision matches user preference if it matches the most favorite option of that user.*

Based on the above assumptions and definition, the adaptation algorithm for RN-Metanetwork is introduced:

Adaptation Algorithm

Step 1: update the meta-layer network

- In Approval case, there is no need to
- In Denial case:

  + Find the most favorite option of each user

$$SV*_i = \text{argmax}(P(S_i)) \quad (5)$$

Where SV*i denotes the most favorite service of user i.

+Count 1 for the value of relevant node related to the user's preference model if the final decision is the same with user's most favorite option. We use the sequential updating [13] as follows:

$$\text{If SV*i = SV*} \text{ then, } P(BN_i \mid \text{Pr}_e) = \frac{P(BN_i).s.q + 1}{s} \quad (6)$$

$$\text{Else, } P(BN_j \mid \text{Pr}_e) = \frac{P(BN_j).s.q}{s} \text{ with } j \neq i$$

With SV*: the finally selected services, Pr_e: the set of evidences of the presences of users, q: the fading factor, $q \in (0, 1)$, s: the effective sample size which is calculated by:

$$s = \frac{1}{1-q} \quad (7)$$

Step 2: update the preference model of each user based on the contribution of that user into the decision

For each user i: Calculate w_i, the distribution probability of that user's preference model: $w_i = P(BN_i)$. Count w_i for the final selected option and update the user's preference model 1 or N times, in approval or denial case respectively.

$$P(S_i = SV* \mid T_e, L_{ei}, CA_{ei}) = \frac{P(S_i = SV* \mid T_e, L_{ei}, CA_{ei}).s.q_i + w_i}{s}$$
$$P(S_i = SV_j \mid T_e, L_{ei}, CA_{ei}) = \frac{P(S_i = SV_j \mid T_e, L_{ei}, CA_{ei}).s.q_i}{s} \quad (8)$$

with $SV_j \neq SV*$

Where T_e, L_{ei}, CA_{ei} denote the evidences of the Time, user location, user current activity of user i, $P(S_i \mid T_e, L_{ei}, CA_{ei})$ denotes the distribution of conditional probabilities in preference model of user i q_i: the fading factor which is calculated separately for each preference model, to maintain the same experience size according to the following equation:

$$q_i = \frac{s - w_i}{s} \quad (9)$$

The probability propagation algorithm and adaptation algorithm which were described above can also be applied for multi-layer Bayesian N-Metanetwork in a recursive manner.The probability propagation process for the Bayesian RN-Metanetwork in Figure 2:

Step 1: applying the probability propagation algorithm for the first 2 layers to calculate the marginal of Used_Pref_BN

Step 2: use the marginal of Used_Pref_BN as the distribution for the Bayesian networks in third layer to calculate the marginal for Service_Category.

The adaptation process for the Bayesian RN-Metanetwork in Figure 2:

Step 1: in case of denial, applying the adaptation algorithm for the first 2 levels of the Bayesian RN-Metanetwork.

Step 2: calculate the marginal of Used_Pref_BN and use it to do adaptation for the third level.

Authorized licensed use limited to: KYUNGHEE UNIVERSITY. Downloaded on November 3, 2009 at 20:11 from IEEE Xplore. Restrictions apply.

VII. ADDING MORE USER PREFERENCE MODELS

When a new user joins the system, the preference model should be updated. The process for integrating a user into a multi-user preference model has 3 steps:

- Create the preference model for that user based on the common preference model for a single user.
- Update the priority model: Add the nodes related to the new user into the priority model, add one more state which is equivalent to the new user preference model and update the CPT as follows:

+ Where new user presence has value NO, the conditional probability can remain the same, and 0 is filled for the probability of the new state.

+ Where the new user presence has value YES, the conditional probability is recalculated. If the system is first-equal-priority the calculation will be:

$$P(BN_N|X,Y,Pr_N=YES) = \frac{1}{N} \quad (10)$$

$$P(BN_i|X,Y,Pr_N=YES) = P(BN_i|X)*\frac{N-1}{N}$$

VIII. CONVERGENCE OF THE ALGORITHM

Practically the adaptation algorithm reaches to a stable state after a few rounds. In this section, we argue about the convergence with the help of game theory.

The system and the users are the players of this game. Assuming that all the players are rational, each user wants to satisfy his/her own preference selfishly and system will want to select a service option which will not be changed by the users (i.e. preferred by the users). We assume there are only two users are present.

So, the dominant strategy of the System should be to activate the Service Option that both the users have agreed upon. This is the Nash-Equilibrium. Probabilistically, the agreed upon service option will be the option with maximum 'weighted sum of preference probabilities of all the users for service options'. The weight is the priority of the user.

But problem is the exact preference probabilities of the user are difficult to learn and the user preferences of the users may change. The online adaptation algorithm adjusts the probabilities appropriately after few rounds.

. In this repetitive game, the objective is to maximize the reward. Here the reward function is the 'weighted sum of preference probabilities of all the users for the activated service option'. It has been proved in [14] that the iterative solution will converge to the equilibrium.

IX. IMPLEMENTATION

The preference model was implemented on a ubiquitous gateway (Ugateway) developed in Ubiquitous Computing Lab. The Bayesian RN-Metanetwork was implemented as an application of ugateway using Hugin java API [15]. The gateway and the metanetwork are deployed as OSGi [16] bundles namely DRCMFramework and DRCMSimulator respectively (see Figure 3). DRCMFramework is the management service of Ugateway that links the sensors, actuators and other services according to the applications' requirements. DRCMBeta.jar is the core set of libraries and APIs needed for the gateway and application. DRCM stands for Dynamic Reconfigurable Context aware Middleware. OSGi enables Ugateway to incorporate various applications in runtime which justify the term Dynamic for the middleware.

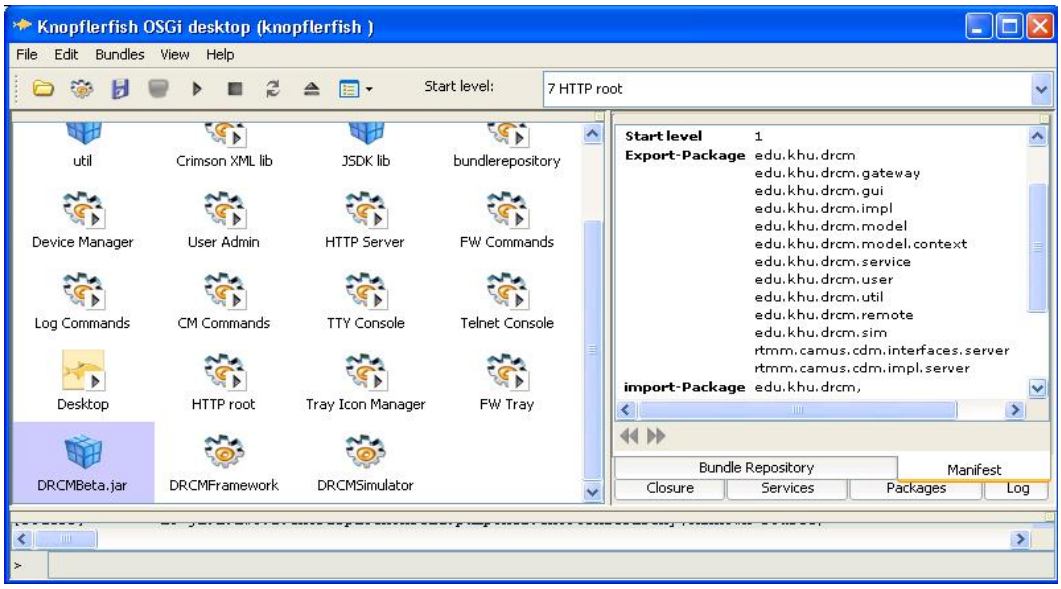

Fig. 3. Ugateway Bundles deployed in OSGI

Authorized licensed use limited to: KYUNGHEE UNIVERSITY. Downloaded on November 3, 2009 at 20:11 from IEEE Xplore. Restrictions apply.

DRCMFramework (Figure 4) has four different functionalities or modules: OWL Model Module, User Provider Module, Gateway Module and Service Module. Each module is managed by a Manager Class, hence there are four managers:

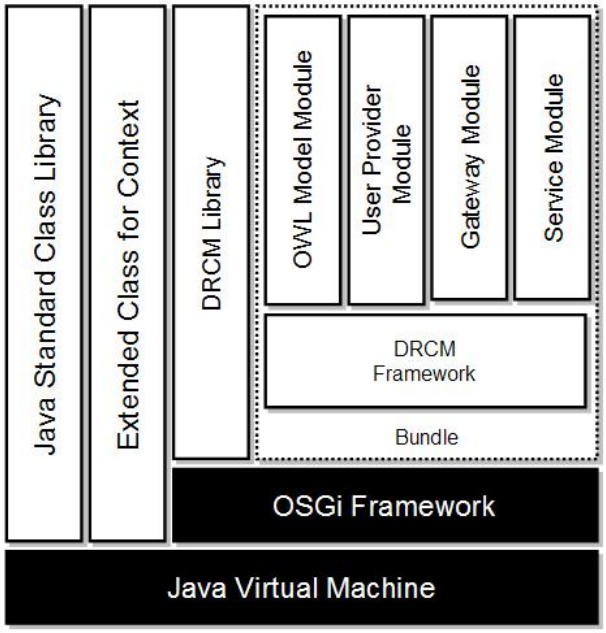

Fig. 4. Architecture of DRCM Framework

ModelManager, UserManager, GatewayManager and ServiceManager.ModelManager models the contexts described in Ontology [17]. UserManager keeps track of the users' context extracted from the context model. GatewayManager manager and ServiceManager are responsible for managing the sensors and services. A DRCM Application should implement itself as GatewayAgent or ServiceAgent or both and then register to the GatewayManager or ServiceManager respectively or both. The GatewayAgent can instantiate a list of sensor contexts by providing ontology and similarly the ServiceAgent can instantiate a list of actuators. The application then can provide updates to the sensor or actuator contexts to the framework and also receive any changes from the ModelManager through loop back methods implemented for GatewayAgent or ServiceAgent within the application. So, the DRCMFramework does not assume any default set of sensors and actuators, rather different applications can instantiate different sets of sensors and actuators. It is possible to provide ontology rules to be acted upon on those contexts or programmer can write complex rules within the application. In our case we implemented Metanetwork as a reasoning mechanism on the context data.

The capability of adding sensors and actuators on the fly by using ontology make Ugateway reconfigurable. The feature was very helpful for our experiment as we could make a simulator of the environment using dummy sensors and actuators. We designed a simulator that models the Professor's room (See Figure 5). The simulator models simple sensors, actuators and the application that tests the convergence of the metanetwork.

User location is taken from which region the user is in the simulator. However, activity is not directly measurable even in the simulator. So, we ran the simulator to collect 500 labeled samples and trained a naïve bayes network for each user to detect his/her activity. The accuracy of this network was around 80%. Activity recognition is still a research topic and researchers have not yet been able to come up with a perfect solution.

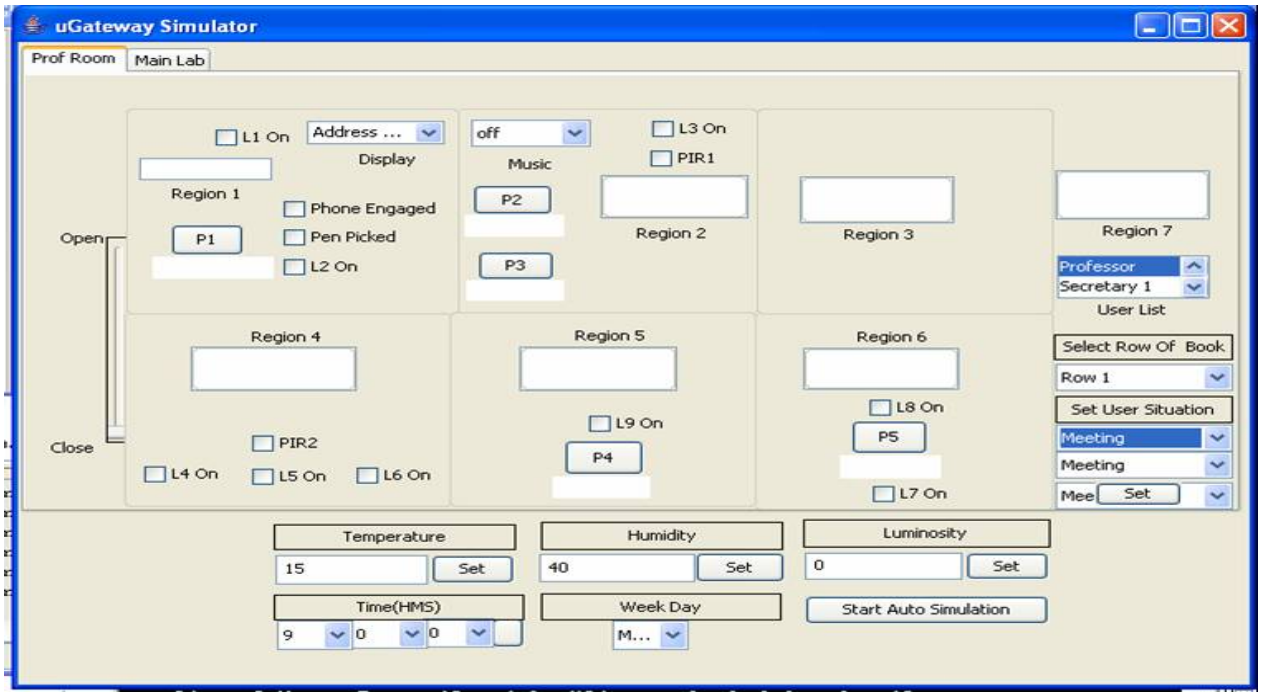

Fig. 5. Ugateway Simulator

Authorized licensed use limited to: KYUNGHEE UNIVERSITY. Downloaded on November 3, 2009 at 20:11 from IEEE Xplore. Restrictions apply.

With the simulator it is possible to generate synthetic user traces inside the professor's room. Three users are modeled: Professor, Secretary1 and Secretary2. There are seven regions; region 7 means outside the room. Professor sits in region 1. There is a pressure sensor (P1) on professor's chair, phone sensor (Phone Engaged), light sensor (L1 on, L2 on), and pen sensor. In region two there are two pressure sensors (P2, P3) on the chairs for visitors, the music player, one light sensor (L3) and a motion sensor (PIR1). Region 3 is empty.

Region 4 has the book shelf with three light sensors (L4, L5, L6) and a motion sensor (PIR2). Region 5 and Region 6 are for secretary 1 and secretary 2 respectively. L7 is the central light sensor. The curtain in behind region 1 and 4. The curtain has only three states, open, half open, close. A particular user can be selected form the text area on the right and then put in Region 7 by clicking on the text area there. Same way the user can be moved from Region 7 to inside the room. User can set the time which is quantized to morning, afternoon and night. Light and Humidity are also quantized into 10 levels each.

The user activity can be set by choosing it from 'User Situation' combo box and clicking on the 'Set' button. Sensor values along with 'User Situation' value produce the labeled data. Once a trace is generated it is saved in a text file and can be repeated by clicking on the 'start auto simulation' button.

The metanetwork is designed for Curtain control, music control, light control. One extra node 'light_condition' has been added to preference models of curtain and light control. 'humidity' node is added to the preference model of music control as humidity affects human mood.

The priority of user for light control (meta layer) also depends on user activity as when any user is resting, the light intensity should be kept low even if any high priority user comes in.

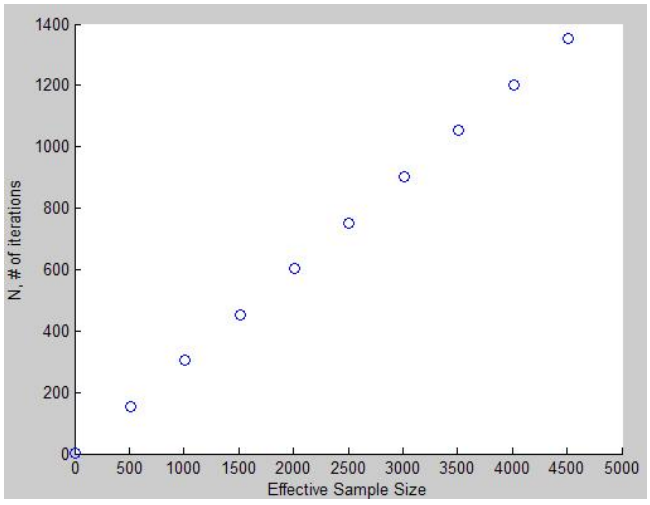

Fig. 6. N vs Effective sample size for adaptation in 3 steps.

X. RESULTS

We took the reaction time of the users to be 30 seconds. Then s and N were determined. In determining s and N, we considered the usual fact that a user will be disturbed if he has to change the system's enabled service option more than three times for a specific service. We took a worst case scenario: two users with metanetwork distribution (0.99, 0.01), two service options with the probability distributions ((0.99, 0.01), (0.98, 0.02)) in the preference model.Suppose the users want the enable service option 2. However, from the current probability distribution it is obvious that the system will enable service option 1. We need to determine for which effective sample size, s and N the system does not take more than 3 iterations to converge to users' decision. It is found that if N equals to around one third of the effective sample size, system converges in 3 adaptation steps (see Figure 6). So, we took an effective sample size of 1000 (fading factor 0.999) and N 300.

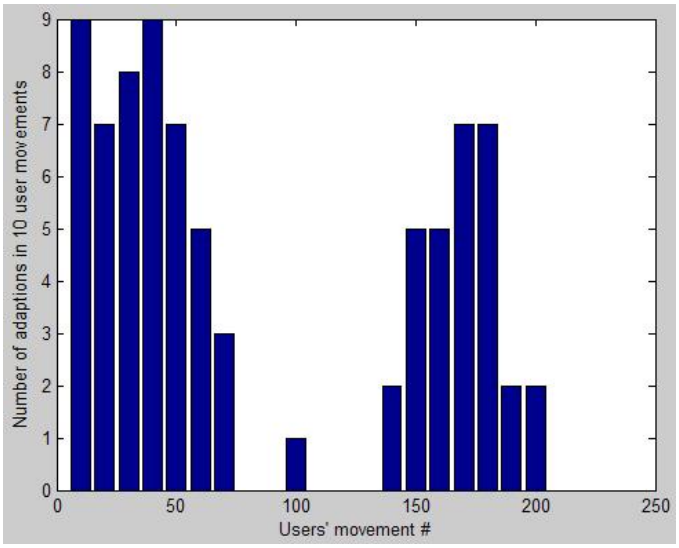

Fig. 7. Number of adaptations in each 10 user movements

However, in reality the adaptation steps may not converge in 3 iterations. Because the user activity recognition algorithm does not have 100% accuracy, the inferred activity may be wrong and hence the system may adapt wrong probability distribution. Again, if the low priority user's favorite options are continued to be selected by the users (changing the system selection), the low priority user's priority increases and may grow higher than the other user). As an effect, the system will enable the low priority user's favorite service option (as he has gained higher priority now) which may be altered. This adaptation procedure continues even after the system becomes stable in the beginning.

We assumed three users (professor, secretary1 and secretary 2) each having equal priority. Each service option also has equal preference. So, initially the priorities of the users along with their preference distribution are adjusted. This is reflected in first 70 movements of the users in Figure 7. From 80~139 movements users repeated their movements and did not change their preferences. However, adaptation of one or two services was not complete before and when the user repeated the movements, adaptation for the service(s) occurred. From 140~169 users changed their preferences drastically which affected the system in later user movements.

Authorized licensed use limited to: KYUNGHEE UNIVERSITY. Downloaded on November 3, 2009 at 20:11 from IEEE Xplore. Restrictions apply.

XI. CONCLUSION

Context-aware computing poses interesting issues for information system researches. Learning user preference in order to adapt the system automatically to the need of user is one of those issues. The challenges in this are due to the uncertain, heterogeneous, distributing characteristic of a context-aware system. Especially when there are many users involve in an intelligent environment, the system has to cope with conflict resolution and distinguishing among the user preferences. A solution for learning user preference in a multi-user context-aware environment which can efficiently resolve the above mentioned problems is the contribution of this paper.

REFERENCES

[1] M. Weiser, "The Computer for the 21st Century," Scientific America, (Sept. 1991) 94-104

[2] Meyer, S. and Rakotonirainy, "A survey of Research on Context-Aware Home," Australasian information security workshop conference on ACSW frontiers 2003, pp. 159-168

[3] Hughes, J., O'Brien, J., Rodden, T., "Understanding Technology in Domestic Environments: Lessons for Cooperative Builds," First International workshop on cooperative buildings (CoBuild'98), Germany, pp. 246-261

[4] W. Keith Edwards and Rebecca E. Grinter, "At Home with Ubiquitous Computing: Seven Challenges.," LNCS 2201, pp. 256-272, 2001

[5] Vagan Y. Terziyan, Oleksandra Vitko, "Bayesian Metanetworks for Modelling User Preferences in Mobile Environment," In proceedings of KI 2003

[6] Nicholas Hanssens, Ajay Kulkarni, Rattapoom Tuchinda, and Tyler Horton, "Building Agent-Based Intelligent Workspaces," In ABA Conference Proceedings, June. (2002)

[7] Choonsung Shin, Woontack Woo, "Conflict Management for Media Services by exploiting Service History and User Preference," ubiPCMM05: 1st International Workshop on Personalized Context Modeling and Management for UbiComp Applications.

[8] Md. Kamrul Hasan, Kim Anh, Lenin Mehedy, Young-Koo Lee, Sungyoung Lee, "Conflict Resolution and Preference Learning in Ubiquitous Environment," International Conference on Intelligent Computing (ICIC 2006), China, pages: 355-366.

[9] Jeonghwa Yang, Bill N. Schilit, David W. McDonald, "Activity Rec-ognition for the Digital Home," IEEE Computer Maganzie, April 08

[10] Gonzalo Huerta Cánepa, Angel Jiménez Molina, In-Young Ko,Dongman Lee, "Adaptive Activity-Based Middleware," In Works in Progress, Activity-Based computing: IEEE Pervasive Computing Magazine, April-June 2008.

[11] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman Publishers, 1979.

[12] Christos H. Papadimitriou and Kenneth Steiqlitz, Combinatorial Optimization: Algorithms and Complexity, Dover Publications; Unabridged edition (January 29, 1998), chapter 15, pages: 342-382.

[13] D. Spiegelhalter, S.L. Lauritzen, "Sequential updating of conditional probabilities on directed graphical structures," Networks, 20:579–605, 1990

[14] Nirmalya Roy, Abhishek Roy, S.K. Das, "Context-Aware Resource Management in Multi-Inhabitant Smart Homes: A Nash H-Learning based Approach," PerCom 2006.

[15] Madsen, A. L. & Kjærulff, U. B. , "Applications of HUGIN to diagnosis and control of autonomous vehicles," in P. Lucas, J. A. Gamez & A. Salmeron (eds), Advances in probabilistic graphical models, Vol. 213 of Studies in fuzziness and soft computing, Springer, pp. 313-332.

[16] OSGi Service Platform, Core Specification, release 4, version 4.1, pages, ISBN 978-90-79350-01-8.

[17] Uschold, M. & Gruninger, M., "Ontologies: Principles, Methods and Applications.," Knowledge Engineering Review, 1996, 11(2).

Md. Kamrul is a Ph.D student in Ubiquitous Computing (UC) Lab, Kyung Hee Univeristy, South Korea. He is also a Senior Engineer at Architect Group, A software security and quality assurance company in Seoul, South Korea since 2008. He received his Bachelors in Computer Science and Information Technology from Islamic University of Technology, Bangladesh in 2002.

Kim Anh Pham Ngoc is working in a software company in Seoul, South Korea. She received MS degree from UC Lab in 2006 and BS from Ho Chi Minh City University, Vietnam. She currently develops software for IPTV.

Young-Koo Lee got his B.S., M.S. and PhD in Computer Science from Korea Advanced Institute of Science and Technology, Korea. He is an Assistant Professor in the Department of Computer Engineering at Kyung Hee University, Korea. His research interests include ubiquitous data management, data mining, and databases. He is a member of IEEE, IEEE Computer Society, and ACM.

Sungyoung Lee received his B.S. from Korea University, Seoul, Korea. He got his M.S. and PhD degrees in Computer Science from Illinois Institute of Technology (IIT), Chicago, Illinois, USA in 1987 and 1991 respectively. He has been a professor in the Department of Computer Engineering, Kyung Hee University, Korea since 1993. He is a founding director of the Ubiquitous Computing Laboratory, and has been affiliated with a director of Neo Medical ubiquitous-Life Care Information Technology Research Center, Kyung Hee University since 2006. He is a member of ACM and IEEE.

Authorized licensed use limited to: KYUNGHEE UNIVERSITY. Downloaded on November 3, 2009 at 20:11 from IEEE Xplore. Restrictions apply.