Conflict Resolution and Preference Learning in Ubiquitous Environment

Md. Kamrul Hasan, Kim Anh, Lenin Mehedy, Young-Koo Lee, Sungyoung Lee[1]

Real-Time & MultiMedia Lab, Department of Computer Engineering,
Kyung Hee University, 449-701, Republic of Korea
{kamrul, anhpnk, lenin}oslab.khu.ac.kr, yklee@khu.ac.kr, sylee@oslab.khu.ac.kr

Abstract. Building intelligent environment is one of crucial challenges for ubiquitous computing developers. To make the environment adapt rationally according to the desire of users, the system should be able to guess users' interest, by learning users' behavior, habit or preference. While learning the user preference, dealing with uncertainty and conflict resolution is of the utmost importance. 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 preference difficult. To address the issue, we propose an approach of user preference learning which can be used widely in context-aware systems. We use Bayesian RN-Metanetwork, a multilevel Bayesian network to model user preference and priority.

Keyword: Conflict resolution, preference learning, Bayesian network, Bayesian RN-Metanetwork, context awareness.

1 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:

[1] Prof. Sungyoung Lee, Ph.D. is the corresponding author.

"*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. Sometimes there is uncertainty in user's temporary preference. User does not always select the most weighted choice. 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. But how the system measures the utility or user satisfaction level? If the system can read the user's expression, and then tell whether the user is sad or happy, or anything else, it makes a very good utility function for the learning process. However, so far this is not realistic. Hence we base the utility function on the user control commands. After the smart system gives a command to an actuator, if the user selects another command, implies he does not agree with the system's decision. If he does nothing, it means he is satisfied. By other words, this is a "Yes/No", or Boolean utility function. Anyway, it is enough for a smart system to learn about the user's preference, at least at the current level. In this paper, we discuss Bayesian RN-Metanetowok [5] for preference learning with conflict resolution scheme. We also discuss the adaptation of the Bayesian model over time span.

To resolve conflicts for Context-aware Media services in smart home environments, GIST [6] 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, this system does not have online learning capability. Reactive Behavioral System (ReBa) resolves conflicts by inferring only the group activities [7], so it is very difficult in ReBa to provide particular services to a specific user when a group of people is present.

In the next section, we point the superiority of our method with an introduction of Bayesian RN-Metanetwork. The subsequent sections describe the modeling of the network for user preference and priority.

2. Bayesian RN-Metanetwork

The smart system should deal with uncertainty, should learn the preference of each user in a multi-user environment, should utilize domain knowledge and user-defined rules and should adapt to the changed user preference. Besides, we see that when combining the preferences of many users, we often base on user priority. Priority can be fixed, but normally it changes over time, or based on situation; for example a sleeping user will be considered more important than others. Hence the context-aware system should be able to learn and adapt to the changes of user's situation-based priority. To address these issues, We propose a Bayesian RN-Metanetwork (RN stands for Relevant Network).

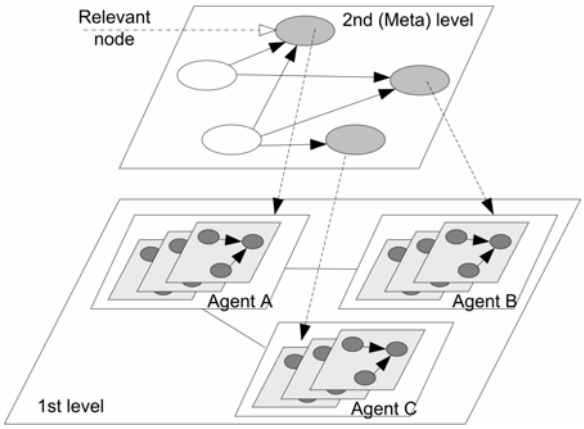

Fig. 1. A Bayesian RN-Metanetwork structure

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:

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

where BNS = {BNS1, BNS2, ... BNSn} is a set of sets of Bayesian networks in first layer and BN0 is the second level 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.
Bayesian RN-Metanetwork supports multi-agent systems. As depicted in Fig. 1, each set of Bayesian networks in the first level is hold by an agent. Each agent uses the distribution of its Bayesian networks to calculate some needed values, and they communicate with other agents through some interfaces. The interfaces consist of common nodes between agents' networks. The mechanism for belief updating in a multi-agent Bayesian network system is described in [8].

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:

- The Bayesian RN-Metanetwork model is very suitable for multi-agent systems. Especially in ubiquitous environments, due to the complex, diverse and open-ended characteristic of the system, the multi-agent paradigm brings much more advantages than the single-agent paradigm.
- The adaptation algorithm for Bayesian RN-Metanetwork is fully described to adapt the model to the continuously changing preference of users.
- Finally, no matter how many users are there in the environment, the priority and preference of each user is calculated separately. This is very useful for widening the scale of systems, as well as knowledge reuse.

3. 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 own preference about a certain service, the last decision to select the service is related to the priorities of the users. Therefore, when modeling the user preference, we also need to model the user priority.

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, i.e. it does not change by time or by place or under any condition. In contrast, situation-dependent priority changes when situation changes, such as when user changes his place or activity.

The user preference in case of situation-independent priority can be modeled using the 2-level Bayesian RN-Metanetwork. Fig. 2 shows the Bayesian RN-Metanetwork for modeling user preference about Multimedia service in a 2-user system.

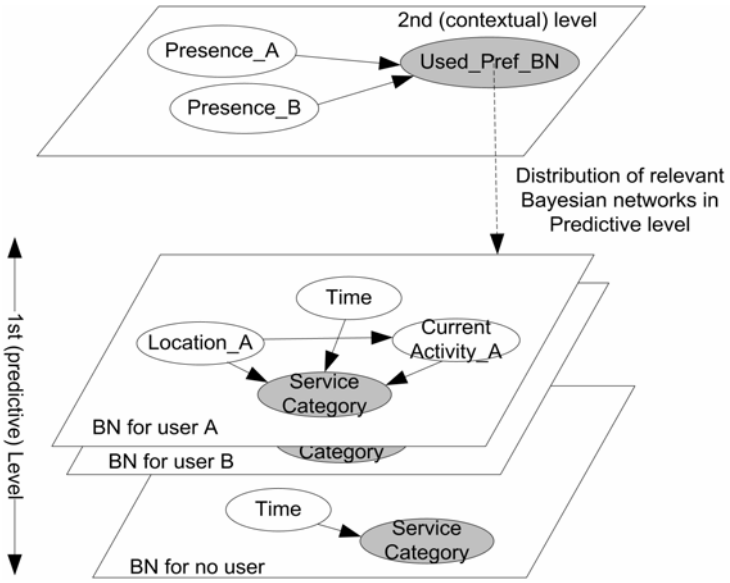

Fig. 2. A Bayesian RN-Metanetwork to learn multimedia service preference of 2 users A, B in case of situation-independent priority.

The model is based on the idea that the priority of one user can be understood as how much that user can contribute into the final decision. In the model, the distribution of relevant Bayesian networks, which model individual user preferences, indicates the proportion of each user preference in the compound preference. Hence the meta-level of this Bayesian RN-Metanetwork also models the user priority. We can see that the user priority in this case depends only on the user presence. Table 1 shows an example of value of the conditional distribution of relevant Bayesian networks (or user priority).

Table 1. Conditional probability of Used_Pref_BN node in 2nd level Bayesian network in Fig.2

Presence_A	Y		N	
Presence_B	Y	N	Y	N
Use_A_pref	0.5	1	0	0
Use_B_pref	0.5	0	1	0
Use_Nouser_pref	0	0	0	1

From the conditional probability table, we see that the two users have same priority. When both users are present, each of them contributes 50% into the final preference decision.

In case of situation-dependent priority, the 2nd (or Meta) level is replaced by a metanetwork. In other words, one more level for learning priority based on situation is added. Fig. 3 depicts the Bayesian RN-Metanetwork for the multimedia services preference in situation-dependent priority case.

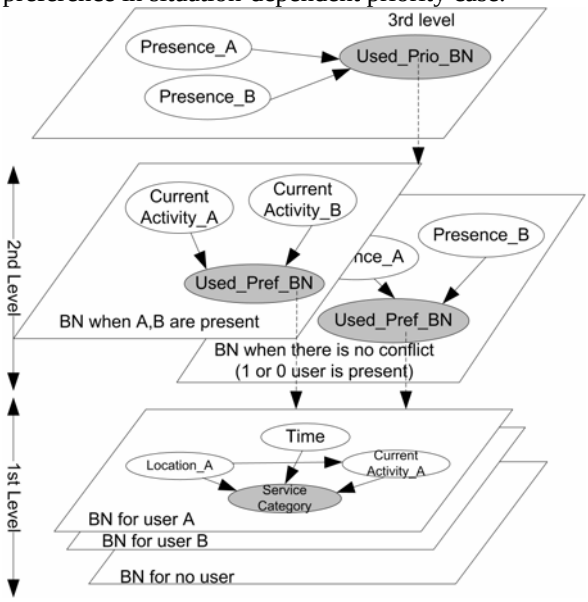

Fig. 3. A Bayesian RN-Metanetwork to learn multimedia service preference of 2 users A, B in case of situation-dependent priority. Priority depends on not only user presence but also user activity.

Table 2. Conditional probability of Used_Prio_BN node in 3rd level Bayesian network in Fig.3

Presence_A	Y		N	
Presence_B	Y	N	Y	N
Use_AB_prio	1	0	0	0
Use_single_prio	0	1	1	1

Table 3. Conditional probability of Used_Pref_BN node when both A and B are present

CurrentActivity_A	Working		Sleeping	
CurrentActivity_B	Working	Sleeping	Working	Sleeping
Use_A_pref	0.5	0.3	0.9	0.5
Use_B_pref	0.5	0.7	0.1	0.5
Use_Nouser_pref	0	0	0	0

Table 4. Conditional probability of Used_Pref_BN node in 2nd level Bayesian network when there is no conflict.

Presence_A	Y		N	
Presence_B	Y	N	Y	N
Use_A_pref	-	1	0	0
Use_B_pref	-	0	1	0
Use_Nouser_pref	-	0	0	1

Tables 2, 3 and 4 show the sample conditional probability of Used_Prio_BN node and Used_Pref_BN nodes. Note that the value range of Current Activity nodes is reduced for easier demonstration (normally we have a lot of activities such as Reading, Walking, Eating, Working, etc.)

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

In this session, we will explain the algorithms for Probability Propagation and Adaptation for the basic 2-layer Bayesian RN-Metanetwork. The algorithms are illustrated by computations on the model in Fig. 2.

Notations and Definitions

The Bayesian RN-Metanetwork in Fig. 2 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 (denoted T), Location_A/B (denoted Li, i=1, 2 corresponding to A and B), CurrentActivity _A/B (denoted CAi), ServiceCategory (denoted 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 (denoted by 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) denotes the probability distribution of the set Pr and $P(Pr_i)$ denotes that of each item in the set.
  - The relevance probability: P (BN). The conditional probability P (BN|Pr)

4.1. Probability Propagation

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

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

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 will be calculated separately, and then combined with the weight coefficients which are the distributed probabilities of the Bayesian networks in Predictive level. The target ServiceCategory SVj with highest P(S=SVj) will be selected.

The Bayesian RN-Metanetwork provides an easy but efficient method for modeling many kinds of user preferences, from multimedia services such as music, television, radio, web page, public information, etc. to the environment parameter such as light, temperature, etc. All we have to do is identify the features which affect the preference of users, build the preference model for each user and finally combine them by a RN-Metanetwork. When the system learns, each preference model is updated separately so that they can be reused in other systems.

4.2. Adaptation

Even though some approaches address learning issue as the initial of the conditional probabilities from example data sets, the true meaning of learning in a ubiquitous system is online learning. When a ubiquitous system starts working, there is no example data but only the domain knowledge and user-defined rules. So, the task of online learning, or adaptation, is crucial.

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. This is a serious case, and the system should not make the same wrong prediction again. So 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. The assumption means that when the system estimates the preference of many users, it gives each user's preference a weight. On the other hand, whenever a control command is given by a group of user, each user has his contribution in that command. The weight in first case and the contribution in second case are assumed to be equal.

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. The assumption means that given the evidences, calculate the posterior distribution of a preference using the preference model of one single user (one of 2 Bayesian networks for 2 users A and B in the previous example, for instance), the option with highest probability can be considered that user's most favorite, and is most likely to be selected by the user himself. 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 update the meta-layer network (users satisfy with the current priority)

+ In Denial case:

- Find the most favorite option of each user

$$SV*_i = \arg\max(P(S_i)) \quad (5)$$

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 introduced in [9]. We perform the following adaptation:

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

Else, $P(BN_j \mid \Pr_e) = \frac{P(BN_j).s.q}{s}$ 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.

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

and

$$P(S_i = SV_j \mid T_e, L_{ei}, CA_{ei}) = \frac{P(S_i = SV_j \mid T_e, L_{ei}, CA_{ei}).s.q_i}{s} \text{ with } SV_j \neq SV^*$$

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 RN-Metanetwork in a recursive manner.

The probability propagation process for the Bayesian RN-Metanetwork in Fig.3:

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 Fig.3:
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.

4. Evaluation

One can argue that the traditional Bayesian network is still faster than the Bayesian RN-Metanetwork, because for Bayesian RN-Metanetwork we have the overhead of the meta-layer propagation. In fact, the Bayesian RN-Metanetwork is slower just in case the distribution of every Bayesian network in the first layer is not equal to 0, or by other words, when all users are present in the system.

Let estimate the calculation time for the Multimedia preference example above when using the traditional Bayesian network and Bayesian RN-Metanetwork. Analysis in [10] shows the runtime for the brute force method of enumeration is $O(q^m)$, where q is the size of the alphabet (in our example: q = number of values for Service Category i for the preference model, or number of users + 1 in the meta network) and m is the number of unknown variables.

We have seen that Pearl's algorithm [11], for the special case of a polytree, has an efficient runtime of $O(Nq^e)$, where e is the maximum number of parents on a vertex. It can be seen that in the case of the turbo-decoding algorithm [12], the runtime is linear in the size of the network, as evidence is propagated a constant number of times.

Here because both the number of unknown variables and the number of maximum parents will increase when the node number increase, we assume that in general the runtime is $O(q^x)$ with x is proportional to the size of the network. The size, S of the network in traditional case is proportional to the number of user. So with N = number of user and q = number of values for Service Category i for the preference model, the propagation time is $O(q^{\alpha N})$ with α is proportional to the number of node in a single user preference model. From Fig.1, the size S' of each preference network in Bayesian RN-Metanetwork remains the same for every user. S' = S / N. We have the propagation time of each preference network is $O(q^{\alpha})$ and the propagation time of meta network in Fig.1 is O (1) given that we know the location of all the users. Then

the numbers of preference model should be propagated, k has the binomial distribution, because this is the distribution of obtaining exactly k (Presence = Yes) out of N trials.

$$P(k \mid N) = \frac{N!}{(N-k)!k!}.p^{k}(1-p)^{N-k} \quad (\mathbf{10})$$

Where p is the mean probability of one user is presented in the location.

The expected value of k is $\mathrm{E[k]} = \mathrm{N.p}$. So, expected propagation time is $\mathrm{O(Npq}^{\alpha})$. We have $\mathrm{O(Npq}^{\alpha})$ < $\mathrm{O(q}^{\alpha N})$. The Bayesian RN-Metanetwork is still more efficient.

In case of situation-dependent priority, the propagation runtime is added with the time for calculating the priority given the evidences about situation. The runtime for each priority model is $\mathrm{O(q}^{\alpha N^{*}})$ with N* is the mean of the binominal distribution of taking k users from N users. We have N* = N.p. Hence the expected propagation runtime for a priority model is $\mathrm{O(q}^{\alpha Np})$. So, the total propagation runtime is $\mathrm{O(Npq}^{\alpha} + \mathrm{q}^{\alpha Np})$. In this case, it is hard to tell whether the Bayesian RN-Metanetwork is faster or slower. However, besides the speed, the advantages of Bayesian RN-Metanetwork come from at least two aspects:

- First, with the division of a large network into small and single-user models, the design of any Bayesian RN-Metanetwork becomes much easier, especially when we have to assign the conditional probabilities for the network.
- Second, we have the separate models for priority and preference. We also have separate models for each user. This dramatically increases the reusability of the models.

5. Conclusion and Future Works

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. We have first presented the Bayesian RN-Metanetwork which can be used to model the user preference as well as user priority for many users, while still maintain separate

preference model for each user. The propagation algorithm showed how to calculate the composite preference of all the users in the system and make decision about the service to provide. To actively and continuously adapt the models to the newest preferences and priorities of the users, the adaptation algorithm for Bayesian RN-Metanetwork was described. However, this is just the first step to make the system intelligent. There are still a lot of challenges such as user behavior routine learning, in which the prediction of the future actions of user is the most important task. We leave that as a future work.

Acknowledgement

This work was supported by IITA Professorship for Visiting Faculty Positions in Korea from Ministry of Information and Communications.

References

1. M. Weiser: Scientifc America. The Computer for the 21st Century. (Sept. 1991) 94-104; reprinted in IEEE Pervasive Computing. (Jan.-Mar. 2002) 19-25
2. Meyer, S. and Rakotonirainy, "A survey of Research on Context-Aware Home.", Proc. of the Australasian information security workshop conference on ACSW frontiers 2003, pp. 159-168. (2003)
3. W. Keith Edwards and Rebecca E. Grinter, "At Home with Ubiquitous Computing: Seven Challenges," LNCS 2201, pp. 256-272, 2001
4. Hughes, J., O'Brien, J., Rodden, T.: Understanding Technology in Domestic Environments: Lessons for Cooperative Builds. In Proceedings of the First International workshop on cooperative buildings (CoBuild'98). Darmstadt, Germany, Heidelberg, Germany: Springer-Verlag, pp. 246-261, (1998)
5. Vagan Y. Terziyan, Oleksandra Vitko. Bayesian Metanetworks for Modelling User Preferences in Mobile Environment. In proceedings of KI 2003
6. 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.
7. Nicholas Hanssens, Ajay Kulkarni, Rattapoom Tuchinda, and Tyler Horton, "Building Agent-Based Intelligent Workspaces," In ABA Conference Proceedings, June. (2002)
8. Yang Xiang. Probabilistic Reasoning in Multiagent Systems - A Graphical Models Approach. Published by the press syndicate of the University of Cambridge. ISBayesian network 0 521 81308 5. 2002
9. D. Spiegelhalter, S.L. Lauritzen. Sequential updating of conditional probabilities on directed graphical structures. Networks, 20:579–605, 1990
10. G.F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. Artificial Intelligence, pages 393-405, 1990
11. D. J. MacKay R. J. McEliece and J. F. Cheng. Turbo decoding as an instance of Pearl's belief propagation algorithm. IEEE Journal of Selected Areas of Communication, pages 140-152, February 1998
12. E. Rodemich R.J. McEliece and J.F. Cheng. The turbo decision algorithm. Proceedings 33rd Allerton Conference on Communications, Control and Computing, 1995