Structured learning of component dependencies in AmI systems

Todor Dimitrov(1), Josef Pauli(2), Edwin Naroska(1), Christian Ressel(1)

(1)Fraunhofer IMS
Finkenstr. 61
47057 Duisburg, Germany
{firstname.lastname}@ims.fraunhofer.de

(2)Universität Duisburg-Essen
Bismarckstraße 90
47048 Duisburg, Germany
josef.pauli@uni-due.de

Abstract

As information and communication technologies are becoming an integral part of our homes, the demand for AmI systems with assistive functionality is increasing. A great effort has been spent on designing and building interoperable middleware solutions to be used as the basis for such system. What is called for, though, is a clear direction in the way uncertainty about acquired knowledge is learnt and employed. This paper presents a probabilistic framework for learning dependencies between components within a home environment. In our approach, the uncertainty is maintained in a probabilistic knowledge base which is automatically built from semantic descriptions and observations of device states and events. The knowledge base can be used by smart applications for performing reasoning about the current flow of system events. Furthermore, some preliminary results obtained from real world data are presented.

1. Introduction

Ambient intelligence promotes the vision of a context-aware, assistive environment that provides value-added services to its inhabitants. One of the most promising applications of AmI is the support of elderly and handicapped people in their daily activities. Thus, the ability to monitor the mental and physical condition of people and to detect emergency situations is what is needed. To be able to do so, the low-level information provided by the ubiquitously installed sensor and actuator devices has to be fused and mapped to a pre-modeled system understanding of user activities, user behavior and even user intentions.

Being in the center of an AmI system, the inhabitant usually acts in a highly indeterministic fashion. Despite of that, however, he or she leaves a discernible fingerprint on the flow of events that are captured by the system. As a result, various inter-component dependencies may arise. For example, some of the devices in the home environment may never be used together or certain of the device events may exhibit a high temporal correlation. What is important though, it is always the case that the degree of uncertainty has to be accounted for. The purpose of this paper is to present an ongoing effort to build a probabilistic framework for the detection and learning of such dependencies.

1.1. Why detect component dependencies?

To give the reader an idea of how the obtained information can be used by AmI applications, two examples are discussed briefly. The first one deals with the problem of error detection and system monitoring in home environments. As more and more heterogeneous systems are being integrated in the contemporary domotic networks, it becomes difficult to identify component malfunctions and their respective causes. For one thing, not all bus systems and protocols provide adequate monitoring functionality, but for another, even if a sensor is behaving "normally", it could be the case that the value of the measured physical variable is erroneous, e.g. a motion detector does not detect motion. To deal with this problem, the different component dependencies can be examined. If, for example, a motion detector event is highly correlated with a light switch event in a given room, then this information can be used by a monitoring application to detect discrepancies in the current event flow.

As a second example consider a user behavior modeling application, which is capable of identifying deviations from the normal (i.e. learnt) way the user interacts with the system. For instance, an increasing count of user-component interactions or an increasing time between several user-generated events that are correlated may be interpreted as a change of behavior. Should the observations remain consistent over a certain period of time, corresponding actions could be taken, e.g. relatives could be notified. The logic of such applications is encoded by using a high level rule-
based language. Its syntax and semantics are beyond the scope of this paper.

In the following, an overview of the related work is given. Subsequently, the different types of information facets and the construction of the probabilistic knowledge base, which captures them, are presented. The paper concludes with the discussion of some preliminary results obtained in a real-world scenario.

2. Related work

In this section, the research that has been relevant for the development of the outlined system is presented. At first, we discuss the automatic construction of probabilistic models. In [8],[7],[4] and [2] several similar approaches to the creation of Bayesian networks from semantic models are discussed. The authors of [7] and [4] use forward-chain rules for defining the structure and probability distribution of the created networks. The work in [8] presents a method for directly extracting Bayesian network structures from ontologies. To do so, properties in the ontology can be connected by the dependsOn predicate. The main difference to all approaches is that in this work, we build a holistic knowledge base such that a clear correspondence between semantic properties and the probabilistic models is established.

The Data Furnace project [6] targets the development of a “general-purpose probabilistic data management infrastructure for pervasive computing”. Similar to the work proposed in this paper, the Data Furnace models temporal, spatial, and causal dependencies between sensor data. The major difference is found in the way such dependencies are extracted, namely we use an user-centric and rule-based approach for building and extracting the probabilistically enriched knowledge, while the Data Furnace relies solely on statistical learning methods.

The work in [5] proposes a framework for a “function/semantic-free” learning of associations between devices and services within an AmI system. The associations are extracted using Hebbian learning from fuzzy rule-based models of the user interactions with the environment. Thus, causal influences between different sets of devices can be detected. The authors show that the learnt associations can be used to trim the rules corpora used by the fuzzy agents and thus to speed up the adaptation and inference cycles.

3. Types of information facets

In the following, the four different types of information facets will be discussed, namely the temporal, spatial, user-related and functional ones. Normally, all facets are somehow related to one another. For instance, knowing a temporal correlation between events can be used for acquiring knowledge about the spatial distance between the devices that generate the corresponding events.

3.1. Spatial facets

The spatial facets are needed, for example, in those cases where the applications have to reason about the neighborhood of system components, their current proximity to the user, points of action, etc. These properties are not always directly readable from the setup of the home infrastructure. For instance, installing a lighting device in one room would have influence on the brightness of the neighboring rooms. In addition, the grade of influence can be estimated by monitoring the behavior of the users, i.e. whenever the user goes into one of the neighboring rooms and he/she does not interact with any of the devices that have direct effect on the luminosity of that room, the brightness could be interpreted as being sufficient.

Another example is presented in figure 1d. It depicts a probabilistic model for learning the sensing area property for several motion detectors, which are installed in a given room. The basic idea is to observe all interactions of the user with devices that are located in the same room and to monitor the events fired by the motion detectors. It is worth mentioning that a single observation of a pair of user interaction and motion detector events is not necessarily sufficient to estimate the inSensingArea property. The problem with single observations is that they could have been caused by multiple people within the home environment. By employing Bayesian networks, we guarantee that the system builds a probabilistic model, which maximizes the MAP es-
timate of the observed data [3], i.e. in the general case, such outliers will be suppressed.

Examples of other spatial facets are: inSameRoom, inNeighboringRoom, adjacetTo, inInteractionRange, etc.

3.2. Temporal facets

The temporal facets capture the information about the time dependencies between observable system events. Temporal information relies extensively on the probabilistic data obtained by Bayesian learning and inference. Like in the case of the spatial facet, it cannot be generally encoded in the domain knowledge base a-priori.

Figure 1c shows a Bayesian network instance, which can be used for learning the correlation between two sensor events. The artificial node deltaT holds the information about the time difference between the two events for each observation. The maximal value of deltaT can be chosen by using either heuristics or the domain ontology. For this purpose, a rich description of the different processes within the home environment (e.g. typical user activities) could be provided. Apart from learning simple time correlations, the framework is capable of creating probabilistic networks for estimating the 13 different interval temporal relations as proposed by Allen et al. [1], e.g. a certain state of one device happens during a given state of another device.

3.3. Functional facets

When learning temporal dependencies, assertions about the logical paths within a home infrastructure are needed. These assertions are provided by the functional information facet. Its main purpose is to describe how devices and even software modules are built up and interconnected. Figure 1a depicts a Bayesian network for learning a functional dependency between components that have an effect on a common physical property within a given room. Since the lamps and the shutters have a functional connection to the luminosity of the room, the states of the lamps could have an influence on the states of the shutters and vice versa. Whether this is the case depends entirely on the provided observations.

Further examples of functional facets are given by causes, dependsOn, requires, etc.

3.4. User-related facets

Reasoning about functional dependencies does not necessarily have to rely only on the provided component models. User preferences and usage patterns also define logical connections that are only observable to the system through the learning process. The whole framework uses a sort of man-in-the-loop reasoning strategy such that the knowledge of how the user behaves (normally) and how he/she interacts with the system is extensively used.

Typical user facets capture the knowledge about the user’s preferences on certain devices (e.g. whether he/she uses only one of the lighting switches in the room), locations (e.g. the time spent in the different locations in the home), etc. In figure 1b, a network, which is used for learning the user’s preference with regard to the lighting switches in a given room, is depicted.

4. Building a probabilistic knowledge base

Learning dependencies between different variables requires well-defined probabilistic models. A popular choice are Bayesian networks as they provide a framework for building interpretations of given influences (causes) in a structured and easily interpretable way. One problem with graphical models, though, is the fact that they are unable to represent several dependency at a time, i.e. the directions of the edges within the network and the way the training data is preprocessed define the entailed semantics. This means that whenever a network is trained from observable data, only certain of the data flavors are going to be captured. Thus, in order to explore all dimensions, several network structures have to be examined.

This is the core idea behind the probabilistic framework that is presented in this paper. Bayesian graphical models are automatically generated from semantic descriptions such that the observable variables are put in different dependency contexts. The structure of the created networks is defined by applying a set of predefined rules. In the following, the construction of a probabilistic knowledge base, which is capable of learning the inter-component dependencies from device event logs, is outlined.

The framework utilizes several knowledge representation formalisms. The basis of the system is formed by the semantic layer. It is comprised of the domain ontology \( O_1(C_1, R_1, A_1) \) (we use a simplified mathematical representation of ontologies; for a complete represen-
A knowledge base $KB_2$ is called empty if it does not define any network models, i.e. $\{BN_1\} = \emptyset$. Otherwise it is a transformation of the instances of $KB_1$ in the domain of $O_2$.

Further, let $I_{C_2\text{node}} = \{I_{C_{2\text{event}}} \cup I_{C_{2\text{state}}} \cup I_{C_{2\text{hidden}}}\}$, i.e. there are Bayesian network nodes that directly correspond to events and states of the devices and there are unobservable nodes, which correspond to any of the other concepts in the domain ontology. Formally, $I_{C_{2\text{event}}}$ and $I_{C_{2\text{state}}}$ are defined as:

$$I_{C_{2\text{event}}} = \gamma_{I_{C_{2\text{event}}}}(\gamma_{I_{C_{2\text{node}}}}(I_{C_{2\text{node}}} \cup I_{C_{1\text{event}}} \cup I_{C_{1\text{state}}} \cup I_{C_{1\text{hidden}}}), I_{R_1})$$

$$I_{C_{2\text{state}}} = \gamma_{I_{C_{2\text{state}}}}(\gamma_{I_{C_{2\text{node}}}}(I_{C_{2\text{node}}} \cup I_{C_{1\text{event}}} \cup I_{C_{1\text{state}}} \cup I_{C_{1\text{hidden}}}), I_{R_1})$$

Definition 3

A Bayesian network semantic model $BN_i$ is called observable (learnable) if it contains event and/or state nodes, i.e. $|I_{C_{2\text{event}}} \cap BN_i| + |I_{C_{2\text{state}}} \cap BN_i| > 0$. In addition, if $I_{C_{2\text{hidden}}} \subseteq I_{C_{2\text{node}}}$, then $I_{C_{2\text{hidden}}}$ is the set of hidden nodes, which are not observable at all times, such nodes should have direct or indirect parents and children (i.e. middle nodes), which belong to the sets of observable nodes. Otherwise, their conditional probability tables can not be learnt by the framework.

In the following example, a simple translation procedure is shown. Figure 2 shows a simplified version of the used domain ontology. It contains the basic concepts for devices, locations and physical characteristics of the environment (e.g. luminosity) as well as the relevant object properties. On the other hand, figure 3 depicts the structure of the Bayesian network ontology $O_2$. The rule set $\gamma_{I_{C_{2\text{node}}}}$ extracts from a sample knowledge base $KB_1$ the PhysicalProperty instances and the DeviceState instances for all devices that have influence of the corresponding physical characteristics. It comprises the following two network node creation rules:

$$\text{PhysProperty}(x) \land \text{genAnnotation}(x, y) \rightarrow \text{InstantiationException}(n, x) \land \text{hasAnnotation}(n, y)$$
PhysProperty(?x) ∧ Device(?d) ∧ effects(?d, ?x) ∧ hasState(?d, ?s) ∧ genAnnotation(?s, ?y) → instantiateNode(?n, ?s) ∧ hasAnnotation(?n, ?y)

The predicate genAnnotation is used for the generation of unique annotations for the created nodes. In this case, the annotations are dependent on the selected physical property of the environment. For the sake of simplicity, the types of network nodes (e.g., binary or multinomial) are left out in the rules’ bodies. This information is rule-dependent and it is normally extracted from the different properties of the concept instances in $KB_1$ that are equivalent to the Bayesian network nodes in $KB_F$. Since the above rules create nodes for the states of the devices that have direct influence of the physical property, the created Bayesian model will be observable and thus learnable from data samples.

Having created the network node instances, what is left is to build the connections between them. This is done with the help of the following rule (the rule builds the $\gamma_{Farc}$ set):

PhysProperty(?x) ∧ Device(?d) ∧ effects(?d, ?x) ∧ hasState(?d, ?s) ∧ genAnnotation(?x, ?y) ∧ getNode(?y) ∧ getNode(?z) ∧ hasParent(?z, ?y)

The predicate getNode is used to retrieved the node with the provided annotation from the created knowledge base $KB_F$. It can be easily seen that all three rules can be merged into a single, more compact representation.

The created Bayesian probabilistic model defines the distribution $P(I_{C_node})$ over the network nodes. The entries of the conditional probability tables are obtained during a learning phase. Since the facet extraction rule-sets put the devices (and their supported states and events) into different dependency situations, it is favorable to have the possibility to control the information flow during the training process. For this purpose, we define the concept of a learning context.

Definition 4 A learning context $\Gamma$ is a filtering function $\Gamma : \{I' = I_{C_{node}} \cap \bar{I}\} \rightarrow \{I'' \subseteq I'\}$ which takes into consideration the values of certain of the device events and states.

As the definition suggests, the learning context takes the observable correspondences of the network nodes in $KB_1$ and returns a subset of them. In this way, certain of the observable nodes can become hidden for the current learning step such that their values are not taken into consideration. If $\Gamma(I') = \emptyset$, then the current system sample is ignored. An example of a learning context is given by the restrictions of the possible system samples (i.e., the values of all observable device events and states) depending on a subset of the sample values. For instance, consider the situation where a facet extraction rule set is defined for learning the influence on the luminosity of the hall from all neighboring rooms. Figure 4 shows an exemplary Bayesian network for learning such a dependency. In this case, it can be favorable to filter the states of the lighting devices such that only state changes caused directly by the user are considered.

Up to now, the structure of the Bayesian networks has been defined and, using the filtered event logs, the system is able to learn the conditional probability tables of the nodes. The next step is to read out the “uncertainty” of the semantic relation. This is done with the help of queries. Let a query be defined as follows:

Definition 5 A query $Q$ is a tuple $(I_{node_{eq}}, I_{node_{eq}} | I_{node_{eq}}, I_{node_{eq}} \subseteq I_{C_{node}} \cap I_{C_{hidden}} | I_{node_{eq}} \cap I_{node_{eq}} = \emptyset)$. The set $I_{node_{eq}}$ contains the queried network nodes (i.e., the result of the query is a distribution $P(I_{node_{eq}})$) and the set $I_{node_{eq}}$ indicates the network nodes, for which evidence can be supplied. Further, the query context $\Upsilon = \{Q_i | i = 1..n\}$ for a given Bayesian network model $BN_i$ defines all allowable queries for the model.

In this way, the query $Q$ divides the observable Bayesian network nodes into query nodes and evidence nodes. Going back to the example with the influence on the luminosity in the hall, an allowable query would separate the nodes in the network for instance into the evidence set $\{L120, L124, L81, L175, L202\}$ and the query set $\{L40\}$ (see figure 4).

As we have seen, the facet extraction rule-sets can be used for defining Bayesian models over a subset of the observable and unobservable instances in the domain ontology. Still, these probabilistic models have to be somehow connected to the semantic relations in the ontology. For this purpose, we formally define the concept of information facets as follows:

Definition 6 An information facet $\Phi$ is a quadruple $(F, \Gamma, \Upsilon, R | R \in R_1)$. It establishes the connection between a property $R$ of the domain ontology and a probabilistic model. The former is the semantic representation...

![Figure 4. A Bayesian network for estimating influence on the luminosity in the hall](image-url)
of a given dependency relation and the later is used for learning and querying the strength of the semantic relation. Further, an information facet instance $\phi$ is a quadruple $(BN_i, \Gamma, \Upsilon, I_R)$. Further, $BN_i \in KB_F, I_R \in I_R_i$.

Information facets can be seen as probabilistically enriched versions of the corresponding semantic relation. They provide the mechanisms for building, training and querying probabilistic models such that the acquired knowledge is semantically consistent with the used relation. Having defined the information facets, it is now possible to construct a probabilistic knowledge base.

Definition 7 A probabilistic knowledge base $\Delta$ is a tuple $\langle \Phi, O_i \rangle$, i.e. it contains a set of information facets, which are defined on the domain ontology $O_i$. Further, an instance $\delta$ of the Bayesian knowledge base $\Delta$ is a tuple $\langle \phi, KB_i \rangle$, which contains the facet instances defined on the knowledge base $KB_i(O_i, I_{G_i}, I_{R_i})$.

The probabilistic knowledge base is the basis of the proposed framework for learning inter-component dependencies. Once constructed, it is automatically trained and adapted. This happens in the so-called inference cycle, which is depicted in in figure 5. From the domain knowledge base, the different information facets are extracted. This is done with the help of OWL-DL and rule reasoning. The purpose of this step is to identify possible dependencies between the devices. To do this, a rich ontological model of the home environment is used. For example, it contains definitions of typical user activities, the requirements on the environment for the normal execution of the activities, etc. Once the information facet instances has been inferred, the corresponding Bayesian models are created and trained from the available observations. Currently, the learning phases are performed at equidistant time instances, taking in consideration the observations from a period of one to several months. It could be the case, though, that the different information facets require different update periods, e.g. time correlation between events can change over time. It is the topic of future research to model such behaviors in the domain ontology.

One of the prerequisites for AmI systems is the ability to adapt to a constantly changing environment. On one hand, this is achieved by the automatically repeated learning phases. On the other hand, the probabilistic knowledge can be transferred back to the domain ontology. Consider again the network in figure 4. If for example L120 is not correlated with L40, this information can be absorbed in the domain knowledge base. In order to determine the grade of influence, we use the expression:

$$c = D_{KL}(P(L40|L120)||P(L40))$$

where the $D_{KL}(P||Q)$ stands for the Kullback-Leibler divergence (cross entropy). If $c < threshold$, than the two distributions are similar, i.e. in this case, the distribution $P(L40|L120) = P(L40)$. For simple cases like the given example, the “weakness” of the dependency can be directly read out from the distribution of the node L40. In more complex situations, where Bayesian network inference is required, the cross-entropy provides a tool for the verification of the separate dependencies within a network. Thus, beginning from a “worst-case” modeling of the inter-component dependencies, the domain knowledge base can be updated incrementally by the framework. Changing certain of the semantic properties within the domain knowledge base could re-trigger another inference cycle, e.g. new instances of the information facets could be extracted.

5. Evaluation

The proposed system has been tested using event logs from 11 different apartments that are occupied by tenants. The used time frame for the analysis has been chosen to be 9 months. All rooms are equipped with lighting actuators and window contacts. In addition, motion detectors are installed in each of the hallways. In the following, the obtained results from the network in figure 4 are discussed. The shown structure corresponds to the layout of apartment 1. For the other apartments, similar networks are constructed.

The network learns how the lights in the rooms, which are connected to the hallway, influence the luminosity in the hallway. Only the device events which occur during the presence of the inhabitant in the hall have been considered (i.e. this is the learning context). Table 1 shows the KL-divergence between the probability distributions for the states of the lighting devices in the hallway given the states of the lighting devices in the neighboring rooms. The distributions are calculated using Bayesian network inference with provided evidence for the Daylight variable and for the lamp state variables for each of the rooms. It can
be seen that in two of the apartments high correlation is present. In the first case (apartment 6), given high luminosity in the living room, the probability for switching the light in the hallway decreases from 56% to 26%. In the second case (apartment 9), the correlation has the opposite sign, i.e. having the lights in the WC switched on causes the probability for switching the lamp in the hallway to increase. The last result can be also interpreted as the hallway being too “dark” in the vicinity of the WC. Since most of the apartments share the same layout, the result shows that it is possible to learn the user perception and/or preferences from observations.

It has also been observed that the different lighting actuators have different grade of influence. Thus, the network can be further used to identify and assign additional semantic properties, i.e. which are the main lights in the rooms, which lights have greater influence on the luminosity in the area of the doors, etc. The discussed example depicts only one possible network structure for detecting component dependencies. Further investigations have revealed that most of the apartments exhibit high correlation between the windows and the lights during the evening hours. This can be explained by the fact that for most people the movement to the window normally “requires” high room luminosity.

These simple conclusions can be highly valuable for the applications that use the knowledge base. They contribute to the discernable fingerprint of the user and are thus fundamental for developing user-adaptable, context-aware AmI solutions.

6. Conclusion and future work

This paper presents a method for the construction of a holistic knowledge base for AmI systems. The knowledge base provides probabilistically enriched information about the different inter-component dependencies within the environment. To bridge the gap between the semantic and probabilistic interpretation of these relationships, the notion of information facets has been introduced. A facet defines the way the dependencies should be extracted from the domain ontology (i.e. the facet extraction rule sets), the way the probabilistic knowledge should be acquired (i.e. the learning context) and the way that knowledge could be transferred back to the semantic domain (i.e. through the query context).

An integral part of the used knowledge inference mechanisms rely on information about the behavior of the home inhabitants. It is the opinion of the authors that user-centric reasoning is an important tool for extracting qualitative information about the relations between the system components. For this reason, the use of rich user and user-component interaction ontologies would be investigated in the future. Such rich models would capture the knowledge of how the user normally behaves, what type of activities he/she performs, what are the requirements on the environment when performing such activities and how these requirements can be used in order to extract possible dependencies between components.

References