Scenario Based Fault Detection in Context-Aware Ubiquitous Systems using Bayesian Networks

Bilal Ahmed, Young-Koo Lee, Sungyoung Lee and Yonil Zhung
Department of Computer Engineering, Kyung Hee University
Sochen-ri, Giheung-eup, Yongin-si, Gyeonggi-do, 449-701, South Korea
bilal@oslab.khu.ac.kr, yklee@khu.ac.kr, sylee@oslab.khu.ac.kr, zhungs@oslab.khu.ac.kr

Abstract

We define the use of Bayesian Networks for fault detection in the perception mechanism of context-aware ubiquitous systems. This paper[1] *describes the complete working of such a fault detection module. Context-Aware ubiquitous systems use a large number of sensors and actuators for their interaction with the environment. The data collected from the environment describes the behavior of the system under different scenarios. If in any way this data or the source of the data gets corrupted then the context formed from such data would be erroneous and result in over-all system misbehavior. Therefore such data needs to be filtered and possible sources of error should be detected in the system's perception mechanism*

1. Introduction

Ubiquitous systems have been designed to facilitate the interaction of humans with computers so that instead of being distinct objects in a user's environment, computers become a part of it by embedding the computations into the environment. Recent work includes making such ubiquitous devices and systems context-aware, enabling the devices or the systems react and adapt to changes which take place in their domain of concern [3]. Achieving context-awareness is not easy because the entire perception of the system is made up of disparate sensors and controllers.

Requirements of context-aware ubiquitous systems include that the system maintain an intense interaction with the environment and make decisions according to the various environmental entities such as users, devices, physical quantities. These decisions are also based on the system itself including the performance of the various software modules and the hardware involved [3]. As a high degree of user ubiquity is needed in such context-aware systems, the system relies heavily on the sensors and controllers it uses to monitor and change the environment. Sensors and controllers are also physical devices which may malfunction under different circumstances. Any such malfunction in the perception mechanism of a context-aware system should not go unnoticed and undetected, so that the higher level context formation remains flawless, keeping the behavior of the system reliable.

Recent research has also tried to make such context-aware ubiquitous systems more autonomous [11]. A system needs to be self-healing, self-reconfigurable, be able to self-optimize and be self-protected [11]. All these concepts require that the system should be able to know the state of its software and hardware at all points. The state of the sensors and actuators which constitute the entire perception mechanism of the system is of vital importance. Knowing the state of the perception mechanism would help an autonomic system to determine the policies required for self-optimization, the current state of the perception mechanism, which quantities/events can be sensed, and in case of possible faults in the perception mechanism the system should be able to carry out isolation or self-healing policies. These requirements stress the need for a fault detection mechanism in context-aware and autonomic ubiquitous systems. This fault detection module should be able to correctly represent the system state at all time and at the same time it should be able to detect any anomaly in perceived data.

In context-aware ubiquitous systems the system possesses enough prior domain knowledge that it can anticipate the changes in the environment and reason about them [3]. This prior knowledge of the domain can be used for sensor fault detection and isolation. The need for fault-detection in the perception mechanism becomes very important for efficient system functionality. The motivation for using Bayesian networks comes from the fact that Bayesian networks not only model variables of a domain but also impose a causal ordering on them [5, 9], and the beliefs of individual variables combine to form the overall belief in the entire modeled system. Scenario based fault detection

[1] This research has been partially supported by KOSEF, Korea

constrains the sensors and actuators to behave in a certain pre-determined fashion. The beliefs of the system in light of the initially sensed data automatically give posterior beliefs about actuator settings and the resulting sensor readings. Thus Bayesian Networks help in determining the state of a certain sensor or actuator based upon acquired sensor data, this is done with the help of prior beliefs about the sensor state and the physical quantity or the event being monitored by the sensor.

2. Context Aware Ubiquitous Systems

Context-aware ubiquitous systems have been designed to maintain continuous interaction with the user and his environment [3]. In doing so the system needs to know the current context in which it is supposed to function. The context is made up of various domain features gathered from sensor and actuator data [3]. These sensors and actuators are susceptible to faults, such faults need to be identified and the erroneous data discarded.

2.1. The Interaction Mechanism

In a context-aware ubiquitous system the entire perception mechanism of a system is composed of a number of diverse sensors deployed in the environment to monitor various physical quantities. A number of controllers or actuators are used by the system to respond to various changes which take place in the environment. The detection of such changes and the formation of context based on these changes is dependent on the data sensed from the monitored environment [3].

In any particular scenario the steps taken by the system can be defined as sensing some data from the environment and acting on its basis. The action taken in the light of the sensed data is determined through various factors such as available resources, the contextual contents, and user preferences. Every such decision step taken by the system also involves sensing data which is needed for validating if the action has indeed succeeded. The complete interaction cycle in a scenario is shown in fig.1.

Therefore sensors are needed both for determining the sequences of changes to be taken by the system and to sense if the desired results have been achieved. Actuators act as the effectors of the system which help it in controlling the state of affairs in the domain of concern.

For detecting anomalies in system-behavior such scenarios need to be identified so that the system behavior becomes predictable, and any deviation from the desired course of action results through some fault at any of the three levels.

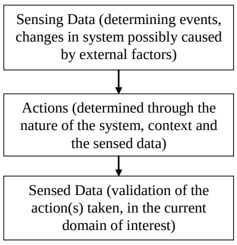
Sensing Data (determining events, changes in system possibly caused by external factors)
Actions (determined through the nature of the system, context and the sensed data)
Sensed Data (validation of the action(s) taken, in the current domain of interest)

Figure 1. The complete interaction cycle of a context-aware ubiquitous system.

2.2. Context Formation

In large context-aware ubiquitous systems, the formation of context plays the most important role in their functionality. Context formation is done, using some prior domain-specific knowledge and the sensed data [3]. Prior domain knowledge can be represented using any feasible knowledge representation technique such as ontology etc [3]. Context is formed by fusing together sensed data and this prior domain knowledge. As this formation of context is done solely on the basis of sensed data, if through any sequence of events the sources of such data get corrupted the context formed would be incorrect. As the contextual knowledge plays the central role in the interaction cycle of a ubiquitous system, incorrect contextual information would result in erroneous system behavior.

3. Fault Detection Using Bayesian Networks

This section outlines a scheme for scenario based fault detection in context-aware ubiquitous system using Bayesian networks. The section outlines the modeling of scenarios, sensors and actuators. At the end of the section the complete network representation and working are explained through an example.

3.1. Modeling a Scenario

The overall system behavior can be monitored for given scenarios which can occur in the domain of concern. This requires the modeling of sensors and the actuators involved in the interaction mechanism of the system. The main reason for using scenarios in modeling system behavior modeling comes from the fact that all sensors and actuators do not collaborate every time, so monitoring and reasoning about the entire set of sensors and actuators becomes computationally infeasible and absurd. Scenarios within the domain of concern pin the current focus of the system on only a subset of all the sensors and actuators, and at the same time

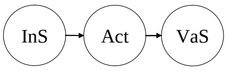

Figure 2. The general scheme for a Bayesian belief network representing a scenario. (InS: Input Sensors, Act: Actuators, VaS: Validating Sensors)

they also define a relation between their behaviors. As explained previously sensors are used by the system at two levels in its interaction mechanism, we need to model them keeping this perspective in mind.

A general Bayesian belief network structure for modeling scenarios is shown in fig. 2. As can be seen from the figure sensors have been modeled at two different levels within the Bayesian network, once for sensing the data, which triggers the response of the system through the actuators, and then again for sensing the desired changes. This structural imposition makes the system more deterministic. The behavior of all the sensors and actuators becomes inter-related given the scenario specifications. This inter-relation helps in detecting any sort of anomaly in the system and isolating any possible faulty component.

As shown in figure 2, a complete scenario is modeled as a serial connection [5, 9], in which the actuator(s) form the connecting node. The model defines a conditional independence between the sensors needed for performing an action (InS) and the sensors for validating the action (VaS), by modeling the actuators as the connecting node. Once the action has been performed the behavior of the sensors in the validating phase is dependent only on the current actuator settings and the initial sensor data need not be considered in the validating phase as depicted by the network structure in fig. 2.

Fig. 2 simply defines an abstract scheme for modeling scenarios, in order to make the network more concrete we need to define the structural representation of sensors and actuators and at the same time the linkages between sensors and actuators also need to be modeled explicitly. These issues are addressed in the subsequent sections.

3.2. Modeling Sensors

For modeling a sensor it is necessary to know its current state. The state of a sensor represents its correctness and can be determined by taking into account certain factors such as the age of the sensor equipment, its reliability as provided by the vendor etc. In order to form a belief network for representing a sensor it is necessary that we take into account the quantity or the event being monitored for example temperature for a heat sensor. The main reason for including the physical quantity in the belief network is purely causal, because it is the physical quantity which causes the sensor to change its value. The belief network should also include the behavior of the sensor as represented by its actual reading, and in the end the state of the sensor should also be a part of the network. These three variables are sufficient to correctly model a sensor. Fig. 3 shows a Bayesian network depicting a sensor.

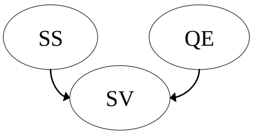

Figure 3. A Bayesian network representing a sensor (SS: Sensor State, QE: Physical Quantity/Event, SV: Actual Sensor Reading)

According to the chain rule for Bayesian networks the joint probability distribution of the model is given by the equation:

$$\begin{aligned} &P(SS, QE, SV) = \\ &P(SV|SS, QE) \times P(SS) \times P(QE) ...(1) \end{aligned} \quad (1)$$

The initial specification of the model includes three potentials namely *P(SS)*: the prior beliefs in sensor state, *P(QE)*: prior beliefs about the monitored quantity, and $P(SV|SS, QE)$ the conditional beliefs about sensor behavior given the sensor state and the monitored quantity.

The model represents a converging connection [5, 9] between the three variables with the actual sensor value being the connecting node. In a converging connection once evidence arrives at the connecting node the other two nodes become dependent. The model is very simple and facilitates the dependence only when actual sensor reading is considered.

Evidence for the model comes in the form of sensor data and is absorbed at the connecting variable. This evidence renders the other two nodes dependent [5, 9], such that based on their prior belief measures their posterior beliefs in light of recent evidence can be computed easily using any of the evidence propagation algorithms for Bayesian belief networks [5, 9]. This means that at any instance a sensor-reading can be used for determining the state of the sensor. The actual sensor reading is entered as an evidence '*e*' into the network, using any algorithm for evidence propagation the posterior beliefs about the sensor state and the physical quantity being measured can be calculated separately.

$$P(SS, QE, SV, e) = P(SS, QE, SV) \cdot e \quad (2)$$

Where (2) represents the absorption of the evidence into the network and (3) shows the belief about the sensor state

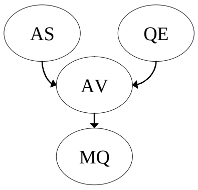

Figure 4. A Bayesian network representing an actuator (AS: Actuator State, QE: Physical Quantity/Event, AV: Actuator Reading, MQ: controlled quantity))

as a consequence of the evidence. The belief measure about the physical quantity can also be calculated using (3) by interchanging *SS* with *QE*.

$$P(SS|e) = \frac{\sum\limits_{QE,SV,e} P(SS,QE,SV,e)}{P(e)} \tag{3}$$

3.3. Modeling Actuators

Actuators are used by the system for controlling various domain objects. As in the case of sensors the actuators should also have some reliability measures. In the current discussion we define actuator-state as being the variable which depicts the current belief in the correctness of the actuator. This parameter can be obtained by considering various factors such as the age of the component, the current environmental conditions, its failure rate as provided by the vendor etc. A Bayesian network designed for an actuator should contain its state, the physical quantity responsible for bringing about a change in the actuator settings, and the physical quantity or the domain object being controlled by the actuator. A Bayesian network depicting an actuator is shown in fig. 4

According to the model shown in figure 4, the controlled object is conditionally independent from the actuator state and the monitored physical quantity, given the actuator configuration. Similarly any evidence on the actuator setting makes the actuator state and the monitored physical quantity dependant on each other. According to the chain rule of Bayesian networks [5, 9] the joint probability distribution of the model can be given as:

$$\begin{aligned}&P(AS,QE,AV,MQ) = \\ &P(MQ|AV) \times P(AV|AS,QE) \times P(AS) \times P(QE)\end{aligned} \tag{4}$$

The model needs prior belief measures about the actuator state, and the physical quantity or event which influences the actuator to change its value. Actuator settings under different environmental conditions are given by the conditional probability measure *P(AV—AS,QE)*. This probability measure also takes into account the current state of the actuator.

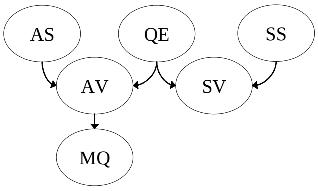

Figure 5. Linking the sensor model to the actuator model.

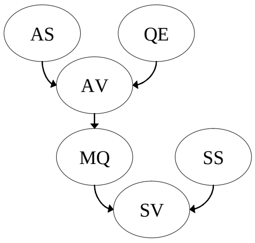

Figure 6. Linking the actuator model to the validating sensor model.

As in the case of the sensor model the state of the actuator can be calculated from the joint probability distribution in light of any evidence '*e*' about the actuator state or about the controlled object, using similar equations

$$P(AS,QE,AV,MQ,e) = P(AS,QE,AV,MQ) \bullet e \tag{5}$$

$$P(AS|e) = \frac{\sum\limits_{QE,AV,MQ,e} P(AS,QE,AV,MQ,e)}{P(e)} \tag{6}$$

3.4. Piecing it Together

We have defined independent models for representing actuators and sensors, but what is needed is a complete model as defined in section 4.1. In order to create a model for a scenario we need to link the models for sensors and actuators. This linking can be done in the light of figure 2. We need only to define the rules for linking together the sensor model with the actuator model at both levels.

At the first level where sensed data is used to perform an action in the domain of concern we need to link together the sensor and actuator models. This linkage is shown in fig. 5.

At the second level, we need to define a linkage between the actuator model and the sensor model for representing the action taken through the actuator and its validation through the sensors. This linkage is shown in fig. 6. From fig. 6 it can be seen that the *QE* node of the original sensor model has been replace by the *MQ*

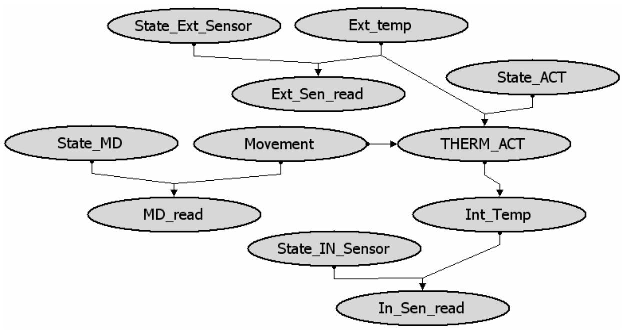

Figure 7. A Bayesian network for the example

node of the actuator. This is because both *QE* and *MQ* are physical quantities or events being monitored by sensors and controlled through actuators.

After defining these linkages we simply need to fuse these two models together and the complete scenario model would be complete. In this completed model actual sensor values would act as the evidence for the complete model. This evidence would then be propagated through the network using any of the algorithms for evidence propagation in Bayesian Networks [5, 9]. In light of this evidence the state of all the sensors and the actuators can be calculated and any possible fault in the system can be detected.

3.5. Example

In this example we present a simple scenario and use the presented technique to come up with the results. The scenario is explained as follows:

"*When there is any user in the room, the internal temperature of the room is adjusted according to the outside temperature. These adjustments or preferences have been pre-fed into the system by the user. The sensors used for this scenario consist of a movement sensor, temperature sensors for external temperature and internal temperature and a thermostat which is used to control the internal temperature of the room.*"

Fig. 7 shows a Bayesian network for the example developed according to the presented technique. Table 1 gives the description of the variables in the network. The set of probability distributions needed for this example is too large to be completely defined here instead we simply provide the prior distributions of the state variables and the physical quantities.

All the state variables have been given the probability distribution of (0.9,0.1) corresponding to the variable state description in table 1. Similarly the prior distribution for movement is given as (0.85,0.15) and for the external temperature it is given as (0.1,0.65,0.15,0.05,0.05) corresponding to the variable state description in table 1.

Probability distribution for one of the sensors is given in table 2, similar probability distributions have also been specified for all other sensors and actuators. The distribution of the actuator reflects the user defined preferences under different circumstances, e.g. the user preference for the internal room temperature if the outside temperature is 31-40 and he is present in the room.

Table 1. Variable Description

Variable Name	State	Description
State_MD*	Correct, Incorrect	State of the movement sensor
MD_read	Yes, No	Reading of the movement sensor
Movement	Yes, No	Prior probability of movement
Ext_Temp	0-10, 11-20, 21-30, 31-40, 41-50	Prior probability of external temperature
Ext_Sen_read	0-10, 11-20, 21-30, 31-40, 41-50	Sensor reading of the external sensor
Therm_ACT	11-15, 16-20, 21-25	Setting of the thermostat
Int_Temp	5-10, 11-15, 16-20, 21-25	Internal temperature
Int_Sen_read	5-10, 11-15, 16-20, 21-25	Reading of the internal temperature sensor

*State_Ext_Sensor, State_ACT, State_IN_Sensor are modeled in the same manner.

Now if evidence is entered into the network in the form of sensor and actuator readings assuming that the internal temperature sensor is malfunctioning we want to see how the network behaves. Let the following set of evidences be entered onto the network:

e1: Ext_sen_read (0,0,0,1,0), e2: MD_read (0,1), e3: THERM_ACT (1,0,0), e4: In_sen_read (0,0,1,0).

The hypothesis variables namely the states of the sensors and the actuators are given as follows:

State_MD = (0.7129 , 0.2871), State_Ext_Sensor = (0.9 , 0.1), State_ACT = (0.9492 , 0.0508), State_IN_Sensor = (0.1854 , 0.8146).

The above hypothesis variables clearly indicate that the internal temperature sensor has malfunctioned, as can be seen from the state variable description of the internal temperature sensor (State_IN_Sensor) which shows that it is incorrect with a belief of 81.46%. This example was simulated using the $MSBNX^{TM}$ tool [12].

Thus, equipped with correct beliefs about all the components in the scenario, some prior knowledge about the state of the components, physical quantities involved this technique would be able

Table 2. Probability Distribution MD_read

State_MD	Movement	MD_read	
		Yes	No
Correct	Yes	1.0	0.0
	No	0.0	1.0
Incorrect	Yes	0.65	0.35
	No	0.35	0.65

to identify any anomaly in the system and its cause on the basis of prior beliefs.

4. Related Work

Fault detection and diagnosis in sensors and sensor networks has been the focus of much research in current years. Some well-established models for fault tolerance in sensors include the celebrated Marzullo model [5] and Iyengar's model [9]. These models have proven very useful in large and distributed sensor networks.

Online fault detection of sensor measurements has also been done using function minimization and non-parametric techniques [4]. The approach uses function minimization and application of non-parametric statistical methods to weed out the most probable faulty sensors in a sensor network. Optimization is achieved by using Powell non-linear function minimization method. Whereas the above mentioned techniques have been applied successfully for fault tolerance and fault-detection in distributed sensor-networks, they do not involve much prior domain knowledge apart from that of the sensors. In a context-aware ubiquitous environment the prior domain knowledge is useful in predicting sensor behavior and modeling complex scenarios which can constrain the behavior of sensors and actuators.

Sensor and actuator Fault detection in large dynamic systems has also been done using stochastic automaton [6]. The addressed systems include those which have discrete valued inputs and outputs. The approach is based on the generalized observer scheme and extends it to deal with discrete valued variables.

Bayesian Networks have been used in fault detection and diagnosis of dynamic systems [7]. The work has been focused on domains related to the control and supervision of large industrial processes involving mixtures of continuous and discrete variables. The main technique in this work includes hybrid dynamic Bayesian networks which capture the stochastic nature of the process and accommodate all type of system variables both discrete and continuous. The application of learning Bayesian networks from system data has also been used for fault detection in large dynamic systems. This method explores the leaning capability of Bayesian networks from measurements of the relevant signals that are present in the dynamic system by the use of a learning algorithm [8]. As opposed to our technique these techniques capture the temporal relations of various process components. Such temporal knowledge is not so critical for the context-aware ubiquitous system to be used efficiently for fault-detection.

Bayesian networks have been successfully used in anomaly detection. Nave Bayesian networks have been employed for detecting anomalies in active networks for providing intrusion detection services [10]. Similarly Bayesian networks have also been used for developing self-aware services which use Bayesian networks to detect any anomaly in their own behavior while functioning on the internet [1].

[2] outlines and classifies the various types of faults which a ubiquitous system can face. It goes on to propose an architecture for a fault manager inside a ubiquitous system. The main focus of the work is on fault-tolerance in large and context-aware ubiquitous systems, dealing with application and device failures. Our proposed scheme deals specifically with the perception mechanism of a context-aware ubiquitous system and addresses in detail the faults which can occur in its hardware components.

5. Conclusion and Future Work

We have defined a technique for fault detection in the perception mechanism of a context-aware ubiquitous system using Bayesian networks. This technique would facilitate the correct context formation based on perceived data, hence improving overall system performance.

The proposed scheme would also be useful in making such context-aware ubiquitous system more autonomic. This scheme can serve to identify system state, thus letting the system know what resources are available, what is the condition of individual components. This particular usage of the scheme in system state identification would help the system in executing various routines for self-healing and self-optimization under different circumstances.

Future challenges include issues relating to efficient storage and identification of scenarios in a context-aware ubiquitous system, and the adaptation and tuning of such Bayesian networks over time for improving their functionality.

References

[1] J. Bronstein, A. Das. Self-aware services: Using bayesian networks for detecting anomalies in internet-based services. *HP Labs Technical Reports HPL-2001-23R1, 2001*.

[2] S. Chetan. Towards fault tolerant pervasive computing. *IEEE Technology and Society*, pages 38–44, 2005.

[3] S. Hung Q. N. Liaquat. Developing context-aware ubiquitous computing systems with a unified middleware framework. In *Proceedings of International Conference on Embedded and Ubiquitous Computing (EUC-2004)*, pages 672–681, 2004.

[4] C. Jacob. Fault tolerance in sensor networks, a survey of fault tolerant sensor network algorithms and techniques.

[5] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer-Verlag, 2001.

[6] M. Koushanfar, F. Potkonjak. On-line fault detection of sensor measurements. In *Proceedings of Sensors 2003 Volume 2*, pages 974 – 979. IEEE, 2003.

[7] R. Lerner, U. Parr. Bayesian fault detection and diagnosis in dynamic systems. In *Seventeenth National Conference on Artificial Intelligence*, pages 531–537. AAAI, 2000.

[8] T. Matsuura, J. P. Yoneyama. Learning bayesian networks for fault detection. In *IEEE Workshop on Machine Learning for Signal Processing*. IEEE, 2004.

[9] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of plausible inference*. Morgan Kauffman, 1988.

[10] T. Sebyala, A. A. Olukemi. Active platform security through intrusion detection using naive bayesian network for anomaly detection. In *Proceedings of LCS 2002*, 2002.

[11] D. Sterritt, R. Bustard. Autonomic computing - a means of achieving dependability? In *Proceedings of the 10th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems*, pages 247–251. IEEE, 2003.

[12] A. Systems and M. R. Interaction Group. Msbnx bayesian network editor and toolkit. *http://research.microsoft.com/adapt/MSBNx/*.