DOI: 10.3745/KIPSTB.2008.15-B.2.137

베이지안 네트워크 기반에 자가관리를 위한 결함 지역화

박 순 선[†] · 박 정 민[††] · 이 은 석[†††]

요 약

결함 지역화는 관찰된 결함의 근본 원인을 자동 인식 하는 것이 가능하기 때문에 규모가 큰 분산시스템에서 중요 역할 수행하며 시스템의 신뢰성 개선을 위해 시스템의 관리와 제어가 가능한 자가 관리를 지원한다. 결함 지역화를 지원하는 기존 연구들은 유비쿼터스 환경에서 베이지안 네트워크와 같은 인공지능 기술들을 주로 사용하여 진단과 예측 기능 중 하나만을 고려하고 있다. 따라서, 본 논문에서는 시스템의 신뢰성 개선을 위해 실시간 시스템 성능 스트림에 대한 학습을 통해 자가관리를 위한 확률적 의존 분석을 기반으로 하는 결함 지역화 방법을 제안하여 진단과 예측기능을 동시 제공한다. 학습 방법으로 베이지안 네트워크 알고리즘을 사용하여 각종 관련된 요소들을 연결함으로써 네트워크를 생성하고 확률적 의존 관계를 통해 귀납적과 연역적 추론기능을 제공한다. 베이지안 네트워크의 구성은 노드들간의 연관성을 찾아내는 것이 중요하기 때문에 그것을 구성하는 인자의 개수가 많은 경우 노드 순서 리스트를 추출하는 사전처리 과정이 필요하다. 따라서 전체 모델링 프로세스에 대한 개선이 요구된다. 이러한 문제를 해결하기 위해 발생한 문제와 관련성이 높은 노드 순서 리스트를 추출하는 방법을 제공한다. 구조 학습을 지원 하는 사전처리 방법을 통해 다양한 문제 영역에서의 학습 효율성을 높이며 학습에 필요로 되는 시간을 줄인다. 제안 방법론을 통해서 시스템의 자원 문제를 신속하고 정확하게 진단하는 것이 가능하며, 관찰된 정보를 기반으로 실행 중에 발생되는 잠재적인 문제를 예측하는 것이 가능하다. 시스템 성능 평가 영역에서 제안 방법론을 적용한 시스템 성능 분석을 기반으로 진단, 예측의 효율성과 정확성을 평가하여 제안 방법론의 유효성을 입증하였다.

키워드 : 결함 지역화, 노드 순서 리스트, 성능 평가, 사전 처리, 확률적 의존성 분석, 자가 관리

Fault Localization for Self-Managing Based on Bayesian Network

Shunshan Piao[†] · Jeongmin Park[††] · Eunseok Lee[†††]

ABSTRACT

Fault localization plays a significant role in enormous distributed system because it can identify root cause of observed faults automatically, supporting self-managing which remains an open topic in managing and controlling complex distributed systems to improve system reliability. Although many Artificial Intelligent techniques have been introduced in support of fault localization in recent research especially in increasing complex ubiquitous environment, the provided functions such as diagnosis and prediction are limited. In this paper, we propose fault localization for self-managing in performance evaluation in order to improve system reliability via learning and analyzing real-time streams of system performance events. We use probabilistic reasoning functions based on the basic Bayes' rule to provide effective mechanism for managing and evaluating system performance parameters automatically, and hence the system reliability is improved. Moreover, due to large number of considered factors in diverse and complex fault reasoning domains, we develop an efficient method which extracts relevant parameters having high relationships with observing problems and ranks them orderly. The selected node ordering lists will be used in network modeling, and hence improving learning efficiency. Using the approach enables us to diagnose the most probable causal factor with responsibility for the underlying performance problems and predict system situation to avoid potential abnormities via posting treatments or pretreatments respectively. The experimental application of system performance analysis by using the proposed approach and various estimations on efficiency and accuracy show that the availability of the proposed approach in performance evaluation domain is optimistic.

Key Words : Fault Localization, Node Ordering List, Performance Evaluation, Preprocessing, Probabilistic Dependency Analysis, Self-Managing

1. Introduction

Self-managing or Autonomic Computing is becoming to be a major issue in distributed system with the rapid growth in size and complexity nowadays especially in ubiquitous computing environment. More requirements in distributed system make more complexities emerge [1], which brings much more burdens and hardness for administrators to handle performance problems and maintain system high reliability. It is very important to system manager for managing the computer system and

※ This work was supported in parts by ITRC IITA-2006-(C1090-0603-0046), Grant No. R01-2006-000-10954-0, Basic Research Program of the Korea Science &Engineering Foundation, and the Post-BK21 Project. Corresponding author: Eunseok Lee
† 준 회 원 : 성균관대학교 전자전기 컴퓨터공학과 석사과정
†† 준 회 원 : 성균관대학교 컴퓨터공학과 박사과정
††† 종신회원 : 성균관대학교 정보통신공학부 교수
논문접수 : 2007년 10월 29일, 심사완료 : 2008년 3월 7일

to users for running their applications successfully. Autonomic computing [2] appears on the field of Information Technology as a challenging topic, which implies that system can recover from faults on its own initiative instead of system administrators' direct handling, for the purpose of providing and maintaining high quality of service without interrupted exceptions. Faults are unavoidable in whole lifecycle of computer systems, and there tasks to remove them immediately for maintaining continuous operations. Therefore, self-managing is urgently required in the evolution of today's autonomic computing systems. Fault localization using Artificial Intelligent techniques generate a variety of challenging applications in an automated way to provide various fault analysis techniques that are applied to diverse fields such as performance evaluation domain.

As increasing complexity in current distributed computing systems, information of system performance is not enough as we need to analyze the root cause of currently observed problems, and the exceptions and abnormalities occur without any anticipation in most instances. Moreover, such uncertainty and noise are the main and unavoidable problems and must be solved in real-life scenarios. Most of existing techniques [3] that focus on analyzing causes or symptoms based on defined rules or cases are not competent in many cases. It is not popular in uncertain domain with missing information and inferring with low accuracy.

Problem localization is a process of deducing the exact root cause of problems based on a set of observed information. Clearly, it is critical to designing an effective self-managing system that determines and solves problems automatically to improve system reliability and quality of service. However, many fault localization techniques use deterministic and probabilistic inference for fault diagnosis and prediction. There are unsolved problems such as overfitting and generalization in recent works. In this paper, we propose an approach to fault localization based on Bayesian network learning to provide probabilistic dependency analysis which is used to localize or predict exact cause of performance problems under given observation in ubiquitous computing environment. We extract node ordering lists that derived from preprocessing course to construct probabilistic dependency model, which improves the efficiency of modeling without degrading the quality of learning. Following the proposed approach in system performance evaluation domain, bidirectional probabilistic inferences are possible to lead to determining location of performance problems and hence achieving self-managing capability. The ability to take appropriate treatments in large-scale computer systems corresponding to inference results brings benefits in improving system reliability.

The rest of this paper is organized as follows. First, we provide related work on various machine learning techniques for problem determination, and list some problems included in existing research. Second, following the introduction of Bayesian network fundamental, we describe in detail the proposed approach to fault localization using Bayesian network, especially in the improved modeling process which include preprocessing. Third, we show a straightforward application using the proposed approach and discuss the implementation of probabilistic dependency analysis for self-managing. At the last section, we give our conclusion of the paper and show the future work.

2. Related work

Self-managing system tasks in Ubiquitous environment such as real-time fault localization and problem diagnosis, call for higher levels of automation. Many recent studies introduce various methods for automated system management [4], attempting to explore new approaches to improve self-managing capability, and hence improve system reliability, such as IBM self-aware distributed systems and Sun fault management in predictive self-healing.

IBM research on self-aware distributed systems aims at automating an increasingly complex and expensive task of real-time problem diagnosis in large-scale distributed system by using state-of-art machine learning - Bayesian inference, probabilistic reasoning and information - theoretic approaches. It shows an architecture of diagnosis system called RAIL (Real-Time Active Inference and Learning), which uses the probe outcomes to make inferences about the system state, and actively requests the next most-informative probes to improve its diagnosis [5]. Thereby, such fault diagnosis is implemented based on assistant cooperation such as intelligent probing techniques. The probing technique imposes a cost, both because of the additional network load which they entail, and also because the probe results must be collected, stored and analyzed.

The Sun Fire X4500 server features the latest fault management technologies. This technology is incorporated into both the hardware and software of the server. Predictive Self Healing introduces a new software architecture and methodology for fault detection, diagnostics, logging, and system service management across Sun's product line. There are two major components in Predictive Self Healing [6]: Fault Management Architecture (FMA) and Service Management Facility (SMF). FMA is a new software infrastructure to stream error detection, diagnostics, recording and fault handling. SMF is a new framework for simplifying management of Solaris system services. Working together, Solaris Predictive Self Healing allows application to proactively respond to a faulty event and correctively perform actions as necessary.

A critical event prediction for proactive management [1] is proposed to build a proactive prediction and control system for large clusters, collecting event logs containing various system reliability, availability and serviceability (RAS) events, and system activity reports (SARs). After the 'raw' system health measurements are filtered, the variables are used for establishing event correlations either through prediction algorithms or root cause solutions using probabilistic networks, including time-series, rule-based classification and Bayesian network models.

Actually, the existing fault localization techniques can be classified into two categories: dependency based methods and non-dependency based methods. However, recent fault localization techniques using machine learning approach such as rule-based or case-based inferences will bring problems because most of them rarely consider relationships between collected information, which are inefficient in the case of uncertainty. 1) The larger the numbers of levels of considering components, the more generated rules are needed, which makes the system experience high overload and low efficiency. 2) All rule or case generations should be user-defined in advanced. 3) All created rules or cases are impossible to be comprehensive, which implies that one event occurred may not be included in the existing aggregation.

Various machine learning techniques are used in the autonomic management of computing system. However, most of them rarely consider mutual relationships between observed parameters. Generally, probabilistic inferences must be done in an environment of uncertainty where the domain information is incomplete and incoming data is uncertain or partially unavailable. In the case of such a situation, with the fact that there are somewhat interrelated relationships between various system metrics, we can start with representing a probabilistic dependency model among system elements rather than deeming them mostly independent in large scale distributed application domains.

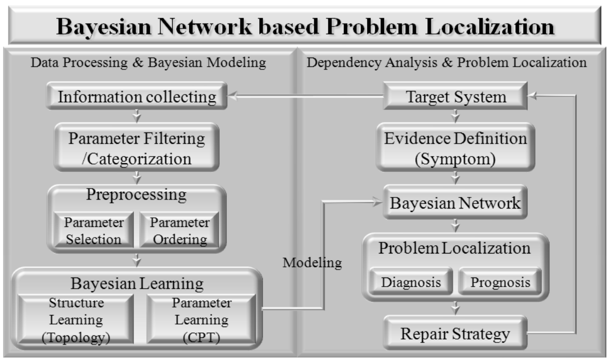
Therefore, we consider the Bayesian network algorithm which is widely used as probabilistic dependency model. Using Bayesian network algorithm to perform fault localization that includes both root cause analysis and proactive problem prediction functions for self-managing system, we should emphasize the method of modeling a compact structure by following an improved process. There are already some works that use Bayesian network for diagnosis or problem localization. A decision support system based on Bayesian Belief Networks was described in [7] to diagnose the problems of a typical enterprise web site that utilizes HTTP Servers, firewalls, messaging servers, and single sign-on. Bayesian networks have been used for developing self-aware services to detect any anomaly in their own behavior while functioning on the internet [8]. As performance problem localization is required to provide self-healing capabilities and deliver the desired quality of service (QoS) in distributed, service-oriented environments, an automated approach to identifying system elements causing performance problems was presented by building on a Bayesian network model that supports probabilistic inference among service elapsed times to end-to-end response time [9]. Bayesian reasoning techniques are applied to perform fault localization in complex communication systems while using dynamic, ambiguous, uncertain, or incorrect information about the system structure and state [10]. However, as we know, structure learning is the main issue when using Bayesian network method. Recently, more researches are induced to learn Bayesian network from data automatically despite that Bayesian network structure can be created by experts based on domain knowledge [11], which is expensive in terms of time and cost, and also manual designed model may be disputed as it is unalterable and unable to reflect to the real-time changes of data. However, most of them are difficult to be carried out in complex domain that should consider a great number of factors, which brings overfitting problem that is one of the main issues in using machine learning. In order to solve the problems that appeared in existing works, we improve the whole process of Bayesian network learning and add the preprocessing which includes parameter selection and parameter ordering to support efficient modeling, and hence overcome the overfitting problems.

3. Fault Localization for Self-Managing

A key essential of self-managing is the ability of the system to perform real-time inferences and learning about its own behavior, to diagnose and predict various problems and performance degradations, namely, the capability of self-awareness. "Suit the remedy to the case". Only with the root cause of a problem can we make the system take appropriate actions or repair strategies to solve the problem. Furthermore, adding proactive prediction ability makes it prevent from unexpected loss through pretreatment, and hence achieve automated system management. Therefore, fault localization based on probabilistic dependency analysis contributes to self-managing for the purpose of determining root causes of problems and predicting problematic situations such as potential problems that going to occur.

3.1 Bayesian Network based Fault Localization Model Structure

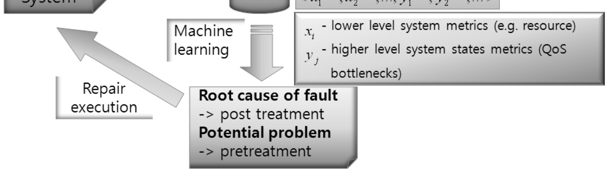
In this paper, we use probabilistic machine learning method, which is mainly used as a modeling tool, to propose a dependency model structure for fault diagnosis and prognosis in self-managing systems. In terms of accuracy and efficiency of diagnosing problems and predicting potential problems, we can deal with the data in the raw beforehand, where the relative parameters in an order are extracted for the next modeling step. Fault localization includes diagnosis and prognosis is executed on the created model via probabilistic dependency inference. It infers the likelihood that a factor is in one state which is dependent on other factors' states that reflect the degrees of confidence. Bayesian network based fault localization model structure for self-managing is described in (Fig. 1) as follows:


Bayesian Network based Problem Localization
Data Processing & Bayesian Modeling
Dependency Analysis & Problem Localization
Information collecting
Parameter Filtering /Categorization
Preprocessing
Parameter Selection
Parameter Ordering
Bayesian Learning
Structure Learning (Topology)
Parameter Learning (CPT)
Modeling
Target System
Evidence Definition (Symptom)
Bayesian Network
Problem Localization
Diagnosis
Prognosis
Repair Strategy

(Fig. 1) Bayesian Network based Fault Localization Model Structure

First, system real-time performance data which also includes the system health states stored in the log file of the target system is collected via monitoring. Second, the information collected from the system log file, is consisting of real-time continuous parameters, should be filtered and discretized. Third, the preprocessing phase, mainly affects the modeling and inference, is processed in advance. Herein we propose an approach based on analyzing information theory among filtered parameters to select a certain parameters that mostly influence on the problematic factors and rank them in an order. Fourth, make Bayesian learning for modeling Bayesian belief network, which includes structure learning and parameter learning. Structure learning finds a network structure that is most probable matching to the training data. Parameter learning decides on the conditional probability table of each node by learning from training data given a constructed network. Fifth, define the degree of confidence information, which is called evidence by presenting as probabilities, including hard evidence and soft evidence. Sixth, post decided evidences to created network to reason out $P(Cause\,|\,Effect)$ or $P(Effect\,|\,Cause)$ in different cases to determine high impact factor of faults or predict potential problems under certain conditions. Finally, determine the parameter with the highest probability in the network after probability propagation when making inferences. According to inference results, it can take corrective repairs on running system in order to keep continuous operation without pause. In order to control system real-time behavior, we collect system real-time health information and detect faults. Through machine learning based on analyzing the information such as low level system metrics and higher level system state metrics, root cause of fault or potential problem is fixed as described at (Fig. 2).

By analyzing statistic data from a given system, we can find patterns without knowing the inner running mechanism and conduct fault localization based on it. We can predict the potential problems beforehand and notify


Target System
Fault detecting
DB
$< x_1^{(1)}, x_2^{(1)}, \ldots, y_1^{(1)}, y_2^{(1)}, \ldots >$
$< x_1^{(2)}, x_2^{(2)}, \ldots, y_1^{(2)}, y_2^{(2)}, \ldots >$
…
$< x_1^{(m)}, x_2^{(m)}, \ldots, y_1^{(m)}, y_2^{(m)}, \ldots >$
Machine learning
x_i - lower level system metrics (e.g. resource)
y_j - higher level system states metrics (QoS bottlenecks)
Repair execution
Root cause of fault
-> post treatment
Potential problem
-> pretreatment

(Fig. 2) Fault Localization based on Probabilistic Inference the alert system to prevent error occurrence, which procedure is called fault prognosis. Fault diagnosis is the opposite process, which analyzes and diagnoses causes of problems under the current system situation. However, both of them are included in the proposed approach based on Bayesian theory, which is used as machine learning for self-managing.

3.2 Fundamental Bayesian Network

Bayesian network is a graphical structure to represent and reason about an uncertain domain, including nodes that represent random variables of interest in the domain and arcs that represent conditional dependencies between pairs of variables. A link between two nodes only implies a direct influence of parent node over child node in the sense that the probability of child node is conditioned on the value of parent node, and two nodes may have a link between them even if there is no direct cause [12]. The formula (1) expressed below is a simple format of Bayes' rule.

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{\sum_i P(B \mid A_i) \cdot P(A_i)} \tag{1}$$

For more complex problems, it also has a mechanism that can propagate probabilities via extending Bayes' Rule throughout the whole network automatically. If a Bayesian network encodes the true independence assumptions of a distribution, we can use a factored representation for the distribution as follows:

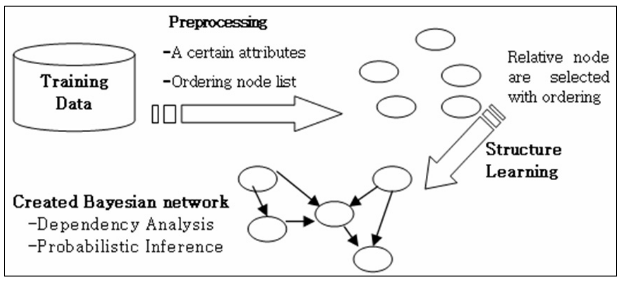
$$\begin{aligned} \mathrm{P}(x_1,\ldots,x_n) &= \prod_{i=1}^{n} \mathrm{P}(x_i \mid x_{i+1},\ldots,x_n) \\ &= \prod_{i=1}^{n} \mathrm{P}(x_i \mid \mathrm{Pa}(x_i)) \end{aligned} \tag{2}$$

Formula (2) shows that instead of the full joint distribution, we need only the conditional probabilities of a variable given its parents, which is based on Markov assumption. A distinct characteristic of Bayesian network is that it is especially useful in uncertainty domains with information about the past and/or the current situation being vague, incomplete, and conflicting. It's easy to explain how a system arrived at a particular recommendation, decision, or action as it can represent probabilistic relationships between nodes dynamically. Furthermore, Bayesian Network can be run in multiple directions, including bottom-up and top-down, which features of Bayesian Network are applied in this paper. Another feature is that it can post evidence to a Bayesian belief network to predict a result or to diagnose a cause based on analyzing current beliefs. The evidence is information about a current situation and beliefs are the probability that a variable will be in a certain state based on the addition of evidence in a current situation [13].

Based on all characteristics of Bayesian network, we can improve the whole process of Bayesian network modeling different from the existing modeling process. A preprocessing course is added to provide mechanism for modeling which is very efficient in the case of considering large number of parameters, and hence solving overfitting problems in the fields of machine learning.

3.3 Preprocessing of Bayesian Network Modeling

Learning Bayesian network from data gives a solution to overcome the limitation of static model created by hand via tracing and analyzing real-time data. Thereby structure learning plays an important part in the whole course and the final results are a direct result of the structure, especially in the case of that having a great number of parameters. The existing researches have begun to investigate methods for learning Bayesian network from data automatically, trying to find the structure that is most probable adapting to the observing information, and it can deal with missing data and hidden variables. The score based search method uses approximate search algorithms to construct candidates and measures them using scoring evaluation. On the other hand, the dependency analysis method starts with analyzing dependency relationships between nodes to construct a network. However, both methods are not suitable and are difficult to be carried out when there are larger data, which brings overfitting that is one of the main issues in using machine learning. The overfitting phenomenon occurs when too many parameters are considered in a given domain. In building Bayesian network structure, it occurs when presenting all parameters in a learned structure. Therefore, in order to solve such problems and make structure learning more efficient, we can provide a preprocessing course for collected information before learning. Given the training data, it selects certain relative factors in an order, and then enters into the next step of structure learning, on which dependency analysis and probabilistic inference are based. (Fig. 3) depicts the brief presentation of preprocessing in Bayesian network learning.



(Fig. 3) Preprocessing in Bayesian Network Learning

When considering too many parameters in structure learning, in order to solve such problems and make structure learning more efficient, we can add preprocessing to improve the whole process of constructing a Bayesian network. Although there are some researches [14][15] on attributes ordering for constructing Bayesian network from databases, the preprocessing in this paper uses different mechanism to extract parameter ordering.

```
Input: Separate observing parameters from problematic parameters stored in set S= {V1 ...Vn, P1...Pm}.
Output: An ordering list with a certain number of parameters
1) Select a certain observing parameters with high relevance to problematic parameters.
  for each problematic parameter Pj ( j= 1 to m) do
      for i = 1 to n do
      compute information gain Gij = Gain( Vi,Pj) = H(Pj)-H(Vi,Pj) (H() means entropy)
      end for
      rank parameters with Gij from maximum to minimum and save them to list Lj
    end for
Combine all lists Lj (j =1 to m), select observing parameters with the mean information gain exceeds
defined threshold value.
Return the selected parameters and all problematic parameters. S'= {V1...Vk, P1...Pm} (k<n)
2) Make an ordering list for the selected observing parameters in set S'
  Initialize set S''={V1,...,Vk} except for problematic parameters; pair set P={empty} ; list L={empty}
  Select two parameters Vx and Vy from the head of set S'' (x!=y)
  Compute Gain(Vx, Vy) and Gain(Vy, Vx) for each pair, put pair(Vx->Vy) with larger Gain into set P
  stop when there is close loop, run until all parameters in set S'' are considered.
  Sort the pairs in set S'' to an single ordering list L
  Return an ordering list L only with observing parameters
```

The course of selecting parameters is described in detail as follows:

First, all the collected parameters are divided into two parts, including observing parameters and problematic parameters.

Second, it computes information gains between a problematic parameter and one of observing parameters.

Third, for each problematic parameter, all computed information gains are ranked from maximum to minimum in a list.

Fourth, combining the gathered lists of all problematic problems, observing parameters with the mean information gain that exceed defined threshold value are selected and returned as a list including all problematic parameters.

Fifth, from the selected observing parameters, two parameters are selected from the head of the list. Then two parameters are stored in a set with order according to the computing results of mutual information by exchanging positions of two parameters. However, the operation is stopped when a close loop appears and continued to run until all parameters are considered.

Sixth, all parameters presented as pairs in the set are realigned to a single list.

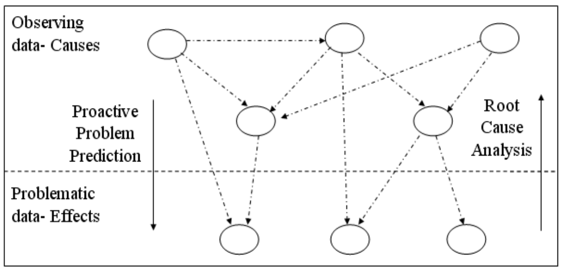
Finally, a certain observing parameters with order are returned as final output.

From the given large dataset with more parameters, it can only choose factors that are more relative with focusing problems, which downsizes the number of factors by using information theory method based on analyzing mutual relationships. From the above course, we can find that problematic parameters are not considered when ranking selected parameters, which implies that all problematic parameters are independent of each other when learning. The anterior parameter may have direct influence on the posterior one in the order, and all observing parameters may have influences on each problematic parameter, because each problematic problem has the same ordering list only with different problematic factor as the last one.

3.4 Methods of Modeling

Recently many methods for structure learning [14] are developed, finding the structure that is most probable to training data. In this paper, a certain parameters with order are taken as input to create a fine-grained model by analyzing conditional independency evaluation between all pairs of nodes. It should be emphasized that the Bayesian network implies conditional independencies via showing conditional probability tables for leaf nodes having direct parent nodes. A structure learning mechanism computes mutual information introduced for pairs of node to reflect different degrees of dependency relationships among them.

In order to build Bayesian network structure for problem localization, it not only to find simple relationships between causes and effects but also to dig out the dependency relations between causes, which enable us to construct a more compact network for problem localization based on probabilistic inferences, as given in (Fig. 4).



(Fig. 4) Bayesian Network Model

With the help of preprocessing of observed parameters, it can determine the direction of arrow in the network when analyzing two nodes have conditional dependency relationship. During other structure learning methods, it can reduce computing complexity and will be efficient especially in the case of much more parameters with large database.

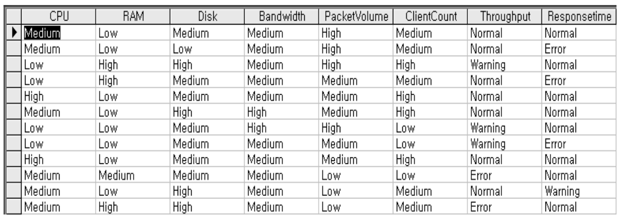
4. Illustration and Evaluation

In rapid growing internet systems in ubiquitous computing era, Service Level Objectives (SLOs) [16] are related to high quality of service such as response time and request throughput as they are a key element of a Service Level Agreement (SLA) between service provider and customer to estimate system reliability. For complex distributed computing system, a huge number of parameters which observed by varied exterior instruments, can represent the whole system status at random time. Fault localization capability in performance evaluation can find which factor of system metrics is directly related to the underlying performance problems by analyzing observed parameters, which consists of performances of individual servers or processes, capability of network, hardware and software, dynamic variation resource utilizations by different type of client requests. Performance parameters such as response time or throughput exceed a certain threshold is also considered as problem or fault. According to the results of probabilistic inferences, the parameters that are responsible for current fault of performance is determined and repaired pertinently. Oppositely, the potential performance problem can be prevented in advance by taking repair strategies.

4.1 Experimental Illustration

First of all, it collects and filters performance data of interest which can be used to analyze from system log files, including utilizations of CPU, memory, disk utilization,

〈Table 1〉 System Performance Metrics



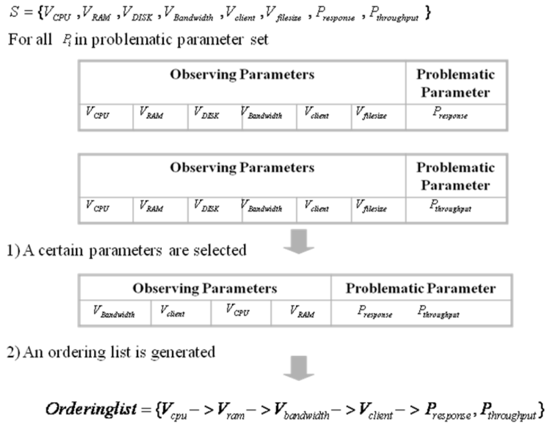
CPU	RAM	Disk	Bandwidth	PacketVolume	ClientCount	Throughput	Responsetime
Medium	Low	Medium	Medium	High	Medium	Normal	Normal
Medium	Low	Low	Medium	High	Medium	Normal	Error
Low	High	High	Medium	High	High	Warning	Normal
Low	High	Medium	Medium	Medium	Medium	Normal	Error
High	Low	Medium	Medium	Medium	High	Normal	Normal
Medium	Low	High	High	Medium	High	Normal	Normal
Low	Low	Medium	High	High	Low	Warning	Normal
Low	Low	Medium	Medium	Medium	Low	Warning	Error
High	Low	Medium	Medium	Medium	High	Normal	Normal
Medium	Medium	Medium	Medium	Low	Low	Error	Normal
Medium	Low	High	Medium	Low	Medium	Normal	Warning
Medium	High	High	Medium	Low	Medium	Error	Normal

bandwidth that logged in a server, and detects information such as threshold violation in response time and throughput. Then, each parameter should be categorized into corresponding classes according to given criteria, as shown in <Table 1>.

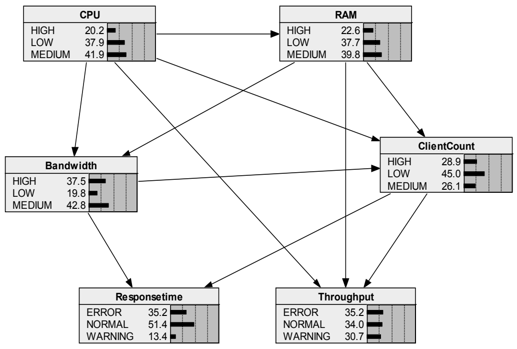
Collected parameters are taken as input to choose the relative parameters in preprocessing which are highly related to occurred SLO violations. Based on learning the training data, observing parameters are ranked by using the proposed preprocessing approach. Finally, it returns a node ordering list containing selected parameters. Referencing to above table, the anterior six parameters are defined as observing parameters and the following two parameters are defined as problematic parameters. In the first step of preprocessing, two observing parameters are removed from the set after analyzing information gains on the training data. Then the remaining parameters are ranked in an ordering list without problematic parameters like this:

$$V_{cpu} \rightarrow V_{ram} \rightarrow V_{bandwidth} \rightarrow V_{client} \rightarrow P_{response}, P_{throughput}.$$

Following the predefined assumption, the problematic parameters response time and throughput are independent of each other. Finally, the returned parameters with order are applied into construct Bayesian network for probabilistic dependency analysis. The flowchart of the preprocessing is shown in (Fig. 5) as follows:


$S = \{V_{CPU}, V_{RAM}, V_{DISK}, V_{Bandwidth}, V_{client}, V_{filesize}, P_{response}, P_{throughput}\}$
For all P_i in problematic parameter set
Observing Parameters
Problematic Parameter
V_{CPU} V_{RAM} V_{DISK} $V_{Bandwidth}$ V_{client} $V_{filesize}$ $P_{response}$
Observing Parameters
Problematic Parameter
V_{CPU} V_{RAM} V_{DISK} $V_{Bandwidth}$ V_{client} $V_{filesize}$ $P_{throughput}$
1) A certain parameters are selected
Observing Parameters
Problematic Parameter
$V_{Bandwidth}$ V_{client} V_{CPU} V_{RAM} $P_{response}$ $P_{throughput}$
2) An ordering list is generated
$Orderinglist = \{V_{cpu} -> V_{ram} -> V_{bandwidth} -> V_{client} -> P_{response}, P_{throughput}\}$

(Fig. 5) Flowchart of Preprocessing

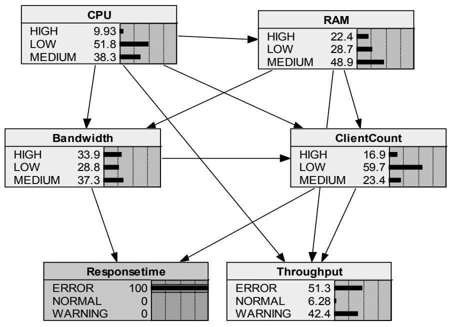


(Fig. 6) Bayesian Network after Learning

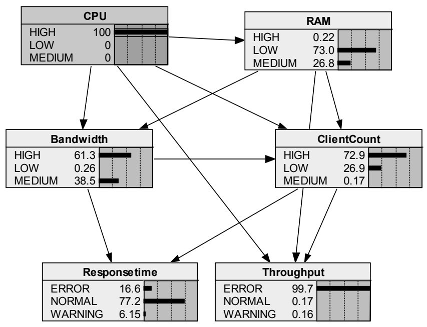
We use Belief Network PowerConstructor software [17] which takes a data set and optional domain knowledge as input and construct a Bayesian network for the data set including both structure and parameters as output. (Fig. 6) describes the created and complete Bayesian network after parameter learning. We can see that the created structure is a compact hierarchy model after learning from certain parameters and ordering list. Different from the simple two-layer structure of Bayesian network, it discovers and represents internal dependency relationships between each pair of causal parameters in the network structure, which makes the model compact and makes the inference results more correct with smaller error rates. After structure learning, parameter learning fixes conditional probabilities for each node under given structure and training data.

Given the convinced states of several parameters, and it makes the known state with assured belief, which operation can change beliefs of all nodes that related to such one node after probabilities being propagated throughout the whole network. As mentioned above, the evidence is information about a current situation, and belief is the probability that a variable will be in a certain state. According to them, we can find the answer which we need by adjusting the beliefs of states of one node, and also can discover that how the nodes affect each other.

For instance, when a violation of response time is observed, which means that it makes response time be of 'error' state, we have the evidence of response time by changing the belief of 'error' state of response time with 100% and the other states with 0%. After then, the most probable impact factor can be decided by finding the 'low' state of one node with max probability comparing to the worse states of other nodes, namely 'low' state of bandwidth. Therefore, we can say that the root cause of response time is bandwidth, and the causal factors can be



(Fig. 7) Bottom-up probability inference for diagnosis



(Fig. 8) Top-down probability inference for prediction

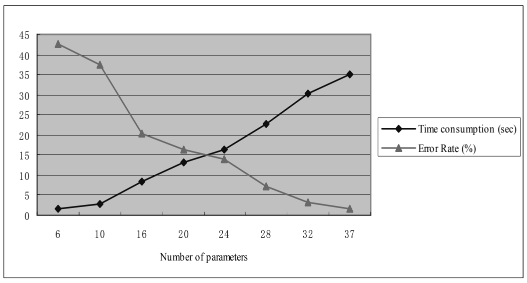
ranked from max probability to min probability of 'low' state. The propagation of probabilities is shown in (Fig. 7).

On the other hand, when the utilization of CPU resource arrives over 95% which means that it belongs to 'high' state, so we adjust the believe of 'high' state of CPU to 100% evidence. Then we can see that the probability of 'error' state of throughput gets up to the max one, which stands for that there will be a fault of throughput appeared in coming time. The changed probabilities propagation is showed in (Fig. 8).

Applying the testing data to the model, the rate of validity is up to over 85%. Thereby, these results derived from probabilistic inferences based on the proposed fault localization using Bayesian network are very helpful for system to take correct repairs to figure out faults or to avoid potential performance faults in advance. From the created Bayesian network model, it's easy for us to understand how the nodes affect each other based on changing the evidences of nodes with dynamic representation.

4.2 Evaluations

For proving the effects of the proposed Bayesian network approach to fault localization for self-managing in performance evaluation, we apply testing data into the built model then compare the results with actual results.



(Fig. 9) Evaluation with different number of nodes

<Table 2> Comparison on cases with and without node ordering

Dimensions	Time consumption (sec)	Accuracy (%)
with node ordering	15.48	90.3
without node ordering	16.37	85.2

<Table 3> Accuracy of Root Cause Analysis

Numbers of Causal Parameters	Root cause in ranking With ordering	Root cause in ranking Without ordering
6	0.9	0.8
10	1.5	1.4
15	1.5	1.6
20	1.7	1.8

<Table 4> Comparisons of Structure Learning Methods in Bayesian Network

	Manual construction	Scoring based method	Dependency analysis method
Requirements	Expert knowledge & Domain knowledge	Scoring measurement & Search algorithm (e.g. Greedy)	Conditional information measurement Domain knowledge
Advantages	Easy to construct network	Efficient for dense structure	Efficient for sparse structure
Disadvantages	Domain experts are lacking; Difficult to reflect real time data dynamically.	Time consuming Computing complexity	Need to define a threshold to decide relationship among nodes
Node Selection	Experts decide relationships	Reduce possible structures	Minimize computing complexity
Node ordering	No need	Reduce search space (select parent of one node before it)	Determine the direction between two nodes

At first, we evaluate time consumption of structure learning and error rate given different numbers of parameters, showing that the obvious effect when using a certain number of parameters that are highly related with the domain. From (Fig. 9), as the number of parameters grows, the time consumption mounts up but the error rate of detecting faults drops, and we can find that the number of parameters corresponding to the crossing of two elements can be chosen as the appropriate quantity of considering parameters in such domain.

Comparisons of time consumption and accuracy are evaluated in the case of selecting certain parameters applying preprocessing or not. <Table 2> can tell us that with node ordering list, there are both improvements on time consumption and accuracy of inference results.

For accuracy of root cause analysis, we estimate the position of the exact cause of current problems in the rankings that include all causal parameters after inferences with different numbers of parameters. Like the results showed in <Table 3>, the average ranks of root causes with ordering are quite close to that without ordering.

As shown in <Table 4>, the comparison of structure learning under given certain quantity of parameters shows that taking an ordering list as input of structure learning can bring high efficiency and accuracy whatever leaning methods are used.

5. Conclusion

In this paper, an approach to fault localization using Bayesian network for self-managing is proposed especially in performance evaluation domain for improving system reliability. In order to improve the performance of learning with domain knowledge, an improved learning process is provided before structure learning. Using the proposed method, we can create a hierarchical network that represents direct relationships between nodes with high efficiency and accuracy, which we use to make probabilistic dependency analysis to determine the exact root cause of system performance problems. Different from other existing researches on using Bayesian network, it adds preprocessing course to extract a certain parameters as a node ordering list for contributing to modeling Bayesian network efficiently. In order to prove the availability and efficiency of proposed approach, we perform it on system performance evaluation domain using the proposed fault localization for self-managing and make comparisons under different conditions.

Since performance management or improvement requires more high level of autonomic functions, we will continue pay attention to using machine learning which is considered as an artificial intelligent approach to learning real time streams of events that expresses the system situations. There are many algorithms, including time-series, decision Tree, case-based reasoning, rule based reasoning that can be integrated for particular mechanism in various fields. It can provide advanced functions by using these methods synthetically.

References

[1] R. K. Sahoo, A. J. Oliner, I. Rish, M. Gupta, J. E. Moreira, S. Ma, R. Vilalta, and A. Sivasubramaniam, “Critical event prediction for proactive management in large-scale computer clusters,” In Proceedings of the ACM SIGKDD, Intl. Conf. on Knowledge Discovery and Data Mining, pp.426 - 435, August 2003.

[2] Jeffrey O. Kephart David M. Chess IBM Thomas J. Watson Research Center, “The Vision of Autonomic Computing,” IEEE Computer Society, January 2003.

[3] Irina Rish, Mark Brodie, Sheng Ma, Natalia Odintsova, Alina Beygelzimer, Genady Grabarnik, and Karina Hernandez, “Adaptive Diagnosis in Distributed Systems,” IEEE Transactions on Neural Networks, March 2005.

[4] Yuan-Shun Dai, “Autonomic Computing and Reliability Improvement,” Proceedings of Eighth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC’05), pp. 204-206, 2005.

[5] IBM Self-Aware Distributed Systems: http://domino.watson.ibm.com/comm/research.nsf/pages/r.ai.innovation.2.html

[6] Sun Microsystems: Predictive Self-Healing in the Solaris 10 Operating System: http://www.sun.com/bigadmin/content/selfheal 0

[7] Bhaskara Reddy Moole and Raghu Babu Korrapati, “Enterprise web site problem diagnosis using Bayesian Belief Networks”, SoutheastCon, Proceedings, IEEE, pp. 384-396, 2005.

[8] 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.

[9] Rui Zhang, Steve Moyle and Steve McKeever, and Alan Bivens, “Performance Problem Localization in Self-Healing, Service-Oriented Systems using Bayesian Networks”, Proceedings of the 2007 ACM symposium on Applied computing, pp. 104-109, 2007.

[10] Malgorzata Steinder, Adarshpal S.Sethi, “Probabilistic Fault Localization in Communication Systems Using Belief Networks”, IEEE/ACM Transactions on Networking, pp.809-822, October 2004.

[11] Jianguo Ding, Bernd Kramer, Yingcai Bai, and hansheng Chen, “Backward inference in Bayesian networks for distributed systems management,” Journal of Network and Systems Management, Vol.13, No. 4, December 2005

[12] Ethem Alpaydm, Introduction of Machine Learning. Massachusetts Institute of Technology, pp.39-60, 2004.

[13] Charles River Analytics Inc, About Bayesian Belief Networks, Charles River Analytics, Inc., 2004.

[14] Jie Cheng, David A. Bell,Weiru Liu, “An algorithm for Bayesian Belief Network construction from Data”, In Proceedings of AI &STAT’, pp. 83-90, 1997.

[15] Cheng, J., Bell, D. and W. Liu, “Learning Bayesian Networks from Data: An Efficient Approach Based on Information Theory”, In Proceedings of the sixth ACM International Conference on Information and Knowledge Management, 1997.

[16] http://www.risi.com/services/sla.html

[17] http://www.cs.ualberta.ca/~jcheng/bnpchlp/index.html

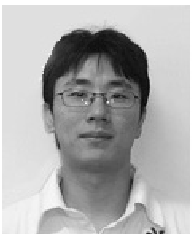
박 순 선

e-mail : sspiao@ece.skku.ac.kr

2004년 7월 중국 대련민족대학교 컴퓨터 공학과(공학사)

2006년 9월~현재 성균관대학교 대학원 전자전기 컴퓨터공학과 (석사과정)

관심분야 : 소프트웨어공학, 오토노믹 컴퓨팅, 자가치유 소프트웨어, 확률 의존 분석, 기계학습



박 정 민

e-mail : jmpark@ece.skku.ac.kr

2003년 2월 한국산업기술대학교 컴퓨터 공학과(공학사)

2005년 2월 성균관대학교 대학원 컴퓨터 공학과(공학석사)

2005년 3월~현재 성균관대학교 대학원 컴퓨터공학과(박사과정)

관심분야 : 소프트웨어공학, 오토노믹 컴퓨팅, 자가치유 소프트웨어, 컴포넌트 기반 개발 방법론

이 은 석

1985년 2월 성균관대학교 전자공학과 (공학사)

1988년 3월 일본 동북(Tohoku)대학교 대학원 정보공학과(공학석사)

1992년 3월 일본 동북(Tohoku)대학교 대학원 정보공학과(공학박사)

1992년~1994년 일본 미쯔비시 정보전자연구소 특별 연구원

1994년~1995년 일본 동북(Tohoku)대학교 Assistant Prof

1995년 3월~현재 성균관대학교 정보통신공학부 교수

관심분야 : 소프트웨어공학, 오토노믹/유비쿼터스 컴퓨팅, 에이전트 지향 지능형 시스템 등