Reasoning Multiagent System for Distributed System Management

Jianguo Ding, Bernd Krämer
Department of Electrical Engineering and Information Engineering
FernUniversität Hagen, D-58084, Germany
{Jianguo.Ding, Bernd.Kraemer}@FernUni-Hagen.de

Abstract. Due to the proliferation of data networks, applications and services, distributed system management have to face the complexity, heterogeneity, uncertainty and dynamic changes in the managed domain. Multiagent technology provides an appropriate method to deal with the full distributed system because of its autonomy and cooperation. On the other side, the probability reasoning is needed for the system management in uncertain environment. In this paper, we consider the collaboration and probability inference between agents for distributed system management. We provide an algorithm of counter inference from effects to causes in Bayesian networks based system management. This is the foundation for further intelligent decision of management system.

1 Introduction

The proliferation of software applications across intranets and the Internet complicates the task of distributed system and network management. In particular, it faces the heterogeneity, complexity, uncertainty and also dynamic changes in network topologies, protocols, system software, application and network services. So an efficient management system should be distributed, automatic, robust and intelligent. Some researchers have already done some work to improve a management system in distributed system. Such as distributed object technology for distributed system and network management [1] [2], web-based distributed system [3], mobile agent [4], intelligent agent based distributed system management [5]. All these research try to improve distributed system management from different requirement. However, it is seldom discussed that how to treat the incomplete information and uncertain situation which is actually inevitable in distributed system management. Meanwhile, most of the current business software, such as IBM Tivoli, HP OpenView, Cisco serial network management software, can give integration, information collection, certain remote monitor, error alarm, and management information statistic. But they still lack of error exactly locating, further information processing and operation automation in uncertain environment. We propose to apply multi-agent systems to automate part of the daily management business.

Agent technology will be an appropriate method to construct a distributed management system because it allows us to assign management tasks to a scalable community of autonomous agents. An individual agent, however, can

This research is involved in IQN (International Quality Networks) project which is supported by DAAD (the German Academic Exchange Service)
only hold local knowledge and lacks a global view of the whole system. When adapting agent technology to distributed system and network management we therefore have to address the following issues:

- Collaboration between agents so that we can integrate their local knowledge effective to obtain a broader basis for decision support.
- The uncertain and incomplete knowledge which comes from the complexity, instability, or unknown factors of the managed system.
- The dependency between the management components or correlated management events.

Bayesian Networks are an effective means to model probabilistic knowledge by representing cause-and-effect relationships among key components of a managed system. Bayesian Networks can automatically generate useful predictions and decisions even in the presence of uncertain or incomplete information.

And in distributed system management, the general operation is trying to inference from the observation of effects to the particular causes. In this paper we develop an algorithm named Strongest Dependence Route for counter-inference in Bayesian networks for multi-agent system.

The rest of this paper is organized as follows: Section 2 presents agent technology, Bayesian networks and its correlated concept. Section 3 discusses the algorithm of Strongest Dependency Route (SDR). Section 4 identifies future research directions.

2 Multiagent Technology and Bayesian Networks

2.1 Multiagent Technology for Distributed System Management

For a distributed system, the information involved is necessarily distributed, and the whole system is complex in some senses, such as: be geographically distributed;

- It has many components;
- It has different management domain based on its management tasks;
- The components of the systems are typically distributed and heterogeneous;
- The topology of the system is dynamic and their content is changing so rapidly that it is difficult for a user or an application program to obtain correct information, or for the enterprise to maintain consistent information.

So it is a reasonable method to divide and rule for complex distributed system management. We import agent technology to improve the management tasks. Agent is character with flexible autonomous action such as [6]:

- Reactivity: agents are able to perceive their environment, and respond in a timely fashion to changes that occur in it in order to satisfy their design objectives;
- Pro-activeness: agents are able to exhibit goal-directed behaviour by taking the initiative in order to satisfy their design objectives;
- Social ability: agents are capable of interacting with other agents (and
possibly humans) in order to satisfy their design objectives.

On the other side, we should deal with the integration and cooperation between individual agents according the belief and goal of the agents. So we defined the collective agents which share the same goal or common task as a multiagent system. This means:

- Multiagent environments provide an infrastructure specifying communication and interaction protocols.
- Multiagent environments are typically open and have no centralized designer.
- Multiagent environments contain agents that are autonomous and distributed, and may be self-interested or cooperative.

In multiagent systems, the agents can function as intelligent application programs, active information resources, and will be knowledgeable about information resources that are local to them, and cooperate with other agents to provide global view of the particular management information.

2.2 Bayesian Networks

Bayesian networks (BNs), known as Bayesian belief networks, belief networks, causal networks or probabilistic networks, are currently the dominant knowledge representation in Artificial Intelligence [7]. Bayesian networks uses graphical structures (Directed Acyclic Graph) to represent probabilistic knowledge. We can define Bayesian networks as a triplet (V, L, P). V is a set of variables (nodes of DAG), L is the causal influence among the variables (directed links between nodes), P is a set of probability distributions: \( P = \{ P(v|\pi(v)) | v \in V \} \); \( \pi(v) \) denotes the parents of node \( v \). The DAG is commonly referred to as the dependence structure of the Bayesian networks.

In Bayesian networks, the information known about one node (i.e., effect node) depends on the information of its predecessor nodes that represent its causes. This relationship is expressed by a probability distribution for each effect node, based on the possible values of its predecessor nodes’ variables. Note that an effect node can also lead into other nodes, where it then plays the role of a cause node. An important advantage of BN’s offer is the avoidance of building huge joint probability distribution tables that include permutations of all the nodes in the network. Rather, only a node’s immediate predecessor’s possible states and their effects on the node are necessary.

Figure 1 shows a Bayesian network for the example of one part of the network management.

In this example, the configuration of server and network overload are the causes for traffic jam, \( p(C=t) \) denotes the probability of the configuration of the server; \( p(T|C) \) denotes the probability of \( T(\text{traffic jam}) \) when \( C(\text{Configuration (server)}) \) happened. In this example, the probability distribution describes
the general character in a long time of the network system, not the system states at particular time. Because the real-time states changes so quickly, and really the real-time states may not work as the foundation of next management decision in this situation. That means the persistent property of the system is important for the system management. And we can get the degree of dependency between the management components from the statistic or from the knowledge of experts.

Due to the knowledge representation of BN, large amounts of interconnected and causally linked data can be represented. Generally speaking:

1. BNs can represent deep knowledge by modeling the functionality of the transmission network in terms of cause and effect relationships between element and network behavior and faults.
2. They can provide guidance in diagnosis. Calculations over the same BN can determine both the precedence of alarms and the areas that need further clarification in order to provide a finer grained diagnosis.
3. They can handle noisy, transient, and ambiguous data due to their grounding in probability theory.
4. They have a modular, compact, and easy to understand representation, when compared to other probabilistic methods.
5. They provide a compact and well-defined problem space because they use an exact solution method for any combination of evidence or set of faults.

BNs are appropriate for automated reasoning because of their deep representations and precise calculations. A BN represents cause and effect between observable symptoms and the unobserved problems so that when a set of evidence are observed the problems most likely to be the cause can be determined. In practice, the network is built from descriptions of the likely effects for a chosen

---

**Figure 1: A Bayesian network for network management example**
fault. In use as a diagnostic tool, the system reasons from effects back to causes.

2.3 Multiply Sectioned Bayesian Networks for Multiagent System Cooperation

In distributed system management, a problem domain may be too complex because of its scale, complexity, heterogeneity, and it will be too difficult to build a single agent to deal with the reasoning task for the entire domain. The problem domain may spread over a large geographical area, and owing to communications cost, delay, and unreliability transmitting observations from many regions to a central location for processing is undesirable.

We use multiagent paradigm to model the reasoning task of a large and complex uncertain problem domain. In the multiagent system, a set of agents can cooperate for particular uncertain reasoning task.

The task of agents is to determine what the true state of the domain is so they can act upon it. The individual agent holds small scale Bayesian network to maintain local knowledge. To determine the states of management event or managed component, each agent much make inferences from its local observations. Because connections between components impose constraints on related event, an agent may benefit from the knowledge and observations of other agents.

So the cooperation between multi agents can be modelled by multiply sectioned Bayesian networks (MSBN). A formal definition of MSBN is as follow [8]: An MSBN M is a triplet (V, L, P): V=∪ 1 1 is the total universe where each 1 is a set of variables called a subdomain. L=∪ 1 1 is the structure where nodes of each subgraph 1 are labelled by element of 1. Let x be a variable and π(x) be all parents of x in MSBN. For each x, exactly one of its occurrences is assigned P(x|π(x)), and each occurrence in other subgraphs is assigned a uniform potential. P=∧ 1 1 is the JPD (joint probability distribution), where each 1 is the product of the potentials associated with nodes in 1. Each triplet Si=(Vi,Li,Pi) is a subnet of M. Two subnets Si and Sj are said to be adjacent if Li and Lj are adjacent in the hypertree.

For large and complex reasoning tasks in multiagent system, some assumptions are given:

1) Each agent’s belief is represented by probability. This assumption not only requires each agent to represent its belief using a probability distribution but also to perform belief updating exactly.

2) An agent communicates directly with another agent only with a concise message: its belief over the variables they share. This basic assumption is consistent with the autonomy of agents. An agent needs to know nothing beyond its subdomain. Because each agent knows nothing beyond its subdomain, the only thing that can be exchange is the belief on the common variables.

3) Each agent represents its subdomain dependence as a DAG (directed
acyclic graph).

(4) The JPD to be consistent with each agent’s belief over its subdomain. The assumption enforces cooperation among agent and interprets the JPD thus defined as the collective belief of all agents. In other words, the collective belief reflects the expertise of each agent within its subdomain and supplements the agent’s limited local knowledge outside its subdomain. The collective belief of the multiagent system, the JPD, is then defined by the product of all such distributions.

According these assumptions, it is possible to integrate the multiply sectioned Bayesian networks into larger Bayesian networks. And every individual Bayesian network is the expression of every agent’s belief.

3 Counter Inference in Bayesian Networks

3.1 Bayes Theory for Inference

Classical inferential models do not permit the introduction of prior knowledge into the calculations. For the rigours of the scientific method, this is an appropriate response to prevent the introduction of extraneous data that might skew the experimental results. However, there are times when the use of prior knowledge would be a useful contribution to evaluation process.

Bayes theorem, developed by the Rev. Thomas Bayes, an 18th century mathematician and theologian, was first published in 1763.

Mathematically it is expressed as:

$$p(H | E, c) = \frac{p(H | c) * P(E | H, c)}{P(E | c)}$$

H: hypothesis;
E: evidence;
c: context;
P(H|E,c): “posterior probability”, the probability of H after considering the effect of E on c;
P(H|c): “prior probability”, the probability of H given c alone;
P(E|H,c): “likelihood”, the probability of the evidence assuming the hypothesis H and the background information c is true;
P(E|c): independent of H and can be regarded as a normalizing or scaling factor.

From this formula, the probability of a Hypothesis can be got given the Evidence. This brings a foundation of counter inference in Bayesian networks.

The most common task in an uncertain reasoning system for distributed system management is probabilistic inference which traces the causes from effects. The task of counter inference amounts to finding the most probable instantiation of some hypothesis variables, given the observed evidence. We define E as the set of effect (evidence) which we can observe, and C as the set of causes. For example, in Figure 2, suppose we observe the state of e0 ∈ E, we think it as an evidence, and now we try to find the most probable causes $C_x \subset C$ and
also the strongest cause $c_i \in C$, which may bring the effects. In this example, we have the assumption that we hold all the probability distribution of $p(v|\pi(v))$ ($v \in V$) for every variable (node) in figure 2.

### 3.2 Strongest Dependency Route (SDR) Algorithm for Bayesian Inference

Before we give the description of the SDR, a simple example is given to demonstrate how the inference is performance in Bayesian networks.

Figure 3 shows a Bayesian network with only 3 nodes. Here we only think of one state of every node. (Actually in a real system, a variable often holds several states.)

Suppose we have the priority information about the Bayesian network: $p(A) = 0.25$, $p(B) = 0.30$, $p(C) = 0.40$, $p(B|A) = 1.00$, $p(C|B) = 0.71$.

Then
\[
p(B|C) = \frac{p(C|B) \cdot p(B)}{p(C)} = \frac{0.95 \times 0.30}{0.40} = 0.71
\]
\[
p(A|B) = \frac{p(B|A) \cdot p(A)}{p(B)} = \frac{1.00 \times 0.25}{0.30} = 0.83
\]

According the belief of the dependency update in every variable, it is easy to trace the inference route and also the dependency relation from $C$ to $A$.

Figure 3 Example of the inference in Bayesian network
As follows, we will give a detail description of SDR algorithm:

(1) Pruning of the BNs

When we dealt with a concrete problem domain, it is common strategy to omit some variables which has no dependency relationship on the problem domain.

Prune BN=(V, L, P), e₀)

\[
\begin{align*}
&v=e₀; V=\{ e₀ \}; L=\Phi; \\
&\text{while } v \neq \text{NIL do} \\
&\quad \{ \\
&\quad \quad V=V \cup \{ \pi(v) \}; \\
&\quad \quad v=\pi(v); \\
&\quad \quad L=L+\langle \pi(v), v \rangle \\
&\quad \} \\
&\}
\end{align*}
\]

For example, in Figure 2 When we get the evidence of e₀, we just want to trace what is the causes of e₀. Figure 4 is the sub-BN from Figure 2 after the Prune operation based on the known effect e₀.
Reasoning Multiagent System for Distributed System Management  

<table>
<thead>
<tr>
<th>i (j)</th>
<th>(j=1)</th>
<th>(j=2)</th>
<th>(j=3)</th>
<th>(j=4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(p(c_j</td>
<td>\Omega))</td>
<td>(Y=0.90, N=0.10)</td>
<td>(Y=0.70, N=0.30)</td>
<td>(Y=0.80, N=0.20)</td>
</tr>
<tr>
<td>(p(a_i</td>
<td>c_j))</td>
<td>(YY=0.90)</td>
<td>(YY=0.85)</td>
<td>(YY=0.95)</td>
</tr>
<tr>
<td></td>
<td>(NN=0.85)</td>
<td>(NN=0.90)</td>
<td>(NN=0.95)</td>
<td>(NN=0.85)</td>
</tr>
<tr>
<td>(p(b_i</td>
<td>a_k))</td>
<td>(YY=1.00)</td>
<td>(YY=0.95)</td>
<td>(YY=0.85)</td>
</tr>
<tr>
<td></td>
<td>(NN=1.00)</td>
<td>(NN=0.90)</td>
<td>(NN=0.95)</td>
<td>(NN=0.95)</td>
</tr>
<tr>
<td>(p(d_i</td>
<td>b_j))</td>
<td>(YY=0.75, N=0.25)</td>
<td>(YY=0.95)</td>
<td>(YY=0.85)</td>
</tr>
<tr>
<td></td>
<td>(NN=0.20)</td>
<td>(NN=1.00)</td>
<td>(NN=0.90)</td>
<td>(NN=0.85)</td>
</tr>
<tr>
<td>(p(e_0</td>
<td>d_j))</td>
<td>(YY=0.95)</td>
<td>(YY=0.95)</td>
<td>(YY=0.95)</td>
</tr>
<tr>
<td></td>
<td>(NN=0.90)</td>
<td>(NN=0.90)</td>
<td>(NN=0.95)</td>
<td>(NN=0.95)</td>
</tr>
<tr>
<td>(p(e_0))</td>
<td>(Y=0.90, N=0.10)</td>
<td>(Y=0.80, N=0.20)</td>
<td>(Y=0.95, N=0.05)</td>
<td>(Y=0.85, N=0.15)</td>
</tr>
</tbody>
</table>

* here YY denotes \(c_i=\text{yes}, a_i=\text{yes}\), NY denotes \(c_i=\text{no}, a_i=\text{yes}\), because 
\(p(a_i=N|c_j=Y)=1-p(a_i=Y|c_j=Y), \ p(a_i=Y|c_j=N)=1-p(a_i=N|c_j=N)\), so we omit YN and NY in this figure.

Figure 5 the probability of every node in Figure 4.

(2) Strongest Dependency Route trace algorithm

In Figure 4 we try to find which the causes for the effect \(e_0\) are and which
the strongest cause is. It is clear that between every cause and effect, they may
have more than one dependency routes. Which route is the strongest dependency
route? And during all the causes, which is the strongest cause?

Before discussing the SDR algorithm, for easy to describe calculation, we
add a virtual super root \(\Omega\) to the BN, we think \(\Omega\) as the necessary event. This
means \(p(\Omega)=1\). So it doesn’t change the property of the whole BN and has no
influence for the result of the inference.

Follow is the description of SDR algorithm which is to trace the strongest
route and cause from effect in Bayesian network.

Algorithm SDR

\{Input:
\ V: the set of nodes (variables) in the Bayesian network
\ L: the set of arc in the Bayesian network
\ P: the dependency probability distribution for every node in Bayesian
\ network
\ s: the initial effect node in BN
\Output:
\ T: a spanning tree \(T\) of the Bayesian network, rooted at vertex \(s\), whose path
\ from \(s\) to each vertex \(v\) is a strongest dependency path from \(s\) to \(v\), and a vertex-
\ labelling giving the dependency probability from \(s\) to each vertex.
\Variables:
\ Depend[v]: the strongest probability dependency between \(v\) and all its
descendants;
\ {Initialize the SDR tree \(T\) as vertex \(s\);
\ Initialize the set of frontier edges for tree \(T\) as empty;
\ Depend[s]=1;
Write label 1 on vertex $s$;
While SDR tree $T$ does not yet span the BN
\{For each frontier edge $e$ for $T$
\{Let $u$ be the labelled endpoint of edge $e$;
Let $v$ be the unlabelled endpoint of edge $e$ ($v$ is one parent of $u$);
\}
Set $P(e) = \text{depend}[u] \cdot p(v \mid u) = \text{depend}[u] \cdot \frac{P(v) \cdot P(u \mid v)}{p(u)}$;
\}
Let $e$ be a frontier edge for $T$ that has the maximum P-value;
Let $u$ be the labelled endpoint of edge $e$;
Let $v$ be the unlabelled endpoint of edge $e$;
Add edge $e$ (and vertex $v$) to tree $T$;
$\text{depend}[v] = p(e)$;
Write label $\text{depend}[v]$ on vertex $v$;
\}
Return SDR tree $T$ and its vertex labels.
\}

In fact, SDR algorithm is try to find a counter route from the effect to its cause, suppose one reasoning serial variables (from cause to effect) is: $x_i \rightarrow \ldots \rightarrow x_{a-1} \rightarrow x_a$, then we can get the joint probability distribution $p(x_a \mid x_{a-1}, \ldots, x_i)$ as the counter reasoning value based on the counter serial variables (from effect to cause): $x_a \rightarrow x_{a-1} \rightarrow \ldots \rightarrow x_i$. It is the ultimate foundation to tradeoff the dependency route from effect to causes. The complexity of SDR is $O(n^2)$, $n$ equals to the size of BN.

Think of the example in Figure 4, the correlated probability dependency is denoted in Figure 5. As an example, a span tree can be generated from effect $p(e_0=N)$ to causes $p(c_i=N)$ by the SDR algorithm. It is showed in figure 6.

![Figure 6](image-url) The span tree of Figure 4 based on effect $e_0$
The strongest path between $e_0$ and every cause and the strongest cause can be get from the span tree. In this example, $R_{i_1} = (e_0, d_1, b_1, a_1, c_1)$, $R_{j_2} = (e_0, d_1, b_1, a_1, c_2)$, $R_{j_3} = (e_0, d_2, b_2, a_3, c_3)$, and $R_{j_4} = (e_0, d_2, b_2, a_3, c_4)$ are the strongest routes. According the priority: Depend[$c_2$] $>$ Depend[$c_3$] $>$ Depend[$c_4$] $>$ Depend[$c_1$], we can inference $c_2$ is the strongest cause when $e_0$ is out of order.

3.3 Related Algorithms for Bayesian Inference

There exist various kinds of Bayesian networks inference algorithms. All in all they can be classified as two kinds of inference: exact inference [9] [10] approximate inference. Each of these offers different properties and works better on different classes of problems, but it is very unlikely that a single algorithm can solve all possible problem instances efficiently. Every resolution is always based on particular requirement. This situation is true for almost all computational problems probabilistic inference using general Bayesian networks has been shown to be NP-hard by Cooper [12].

Compare with other algorithms, SDR algorithm is exact inference from effect to causes and it is easy to trace the dependency route for error detection. Some algorithms, such as clique-tree algorithm [13], which is the most populated exact BN inference algorithm, considers the evident propagation by cluster in BN, it is difficult to trace particular cause from particular effect.

4 Conclusions and Future Work

Multiagent technology is an appropriate method to deal with the complex and distributed system management because of its flexible autonomous and cooperation. It brings a distributed framework for distributed system management. In real distributed system, the unstable, uncertain and incomplete information should be concerned. Fortunately Bayes theory provides a scientific foundation to deal with probability of real system. So it is reasonable using Bayesian networks to represent the belief of agent and model the probabilistic reasoning between agents. Bayesian inference is the popular operation for a reasoning system. The SDR algorithm presents an efficient method to trace the causes from effects in exact inference. It is useful for systems diagnose and error exact location. Most of the distributed system, however, is dynamic update from its structure, topology and its dependency between management components.

Due to the special requirements in distributed systems and network management, we have to concern the following topics:

1. To accommodate sustainable changes in a management system, we investigate learning strategies that allows us to modify the cause-effect structure and also the dependency between the nodes of a Bayesian networks correspondingly.

2. To maintain a health system, it is more important to recognize the potential upcoming system defects than to recover the damaged system.
References


