

Theme 4: Distributed software agents for network fault management

Abstract

This research proposes a framework for distributed management of network faults by software agents. Intelligent network agents with advanced reasoning capabilities address many of the issues for the distribution of processing and control in network management. The agents detect, correlate and selectively seek to derive a clear explanation of alarms generated in their domain. The causal relationship between faults and their effects is presented as a Bayesian network. As evidence (alarms) is gathered, the probability of the presence of any particular fault is strengthened or weakened. Agents having a narrower view of the network forward their findings to another with a much broader view of the network. Depending on the network's degree of automation, the agent can carry out local recovery actions.

1. Introduction

Networks and distributed processing systems are of growing importance and have become an important substrate of modern information technology. The trend in networking is towards high speed communication networks supporting more services and users. Network Managers are faced with the challenge of keeping up with a rapidly changing heterogeneous mix of evolving technologies – hardware and software. There is also an increasing demand to ensure continued network availability or at worst to reduce, as much as possible, outage periods, as this may be critical to the function of organizations. The distributed nature of the network and the amount of information that is present in a network make Intelligent Agents a natural solution to the problem of automating fault detection, correlation and correction.

In this research, a framework for automating fault management is presented. The management function is distributed among multiple agents capable of advanced reasoning activities in their domain.

2. Background and Motivation

The Internet Management Standards defines a network management framework for TCP/IP-based internets [3]. The framework includes the simple network management protocol (SNMP), a structure of management information, and a management information base (MIB). The organization model of the SNMP clusters all active entities in the network in two types: Network Management Stations (NMS) and agents. In Internet the details of *objects*, reflecting the static and dynamic states of managed resources, are stored in a logical database known as the MIB. This management information can be accessed via SNMP and is available at each manager and agent node. To maintain a sufficiently accurate view of the

network, the NMS may need to poll agents frequently for different network statistics. The agents are simple software processes that live on every managed node, collecting, forwarding and setting MIB entries, either at predefined intervals or when requested to by the NMS. The NMS and SNMP agents communicate using the SNMP communication capabilities.

With the increasing amount of information that SNMP agents maintain (could be as much as 500 MIB entries), the bandwidth and computation time consumed during transfer and processing of such bulky data is a concern. Robust management of the network is difficult to achieve. Sometimes, having too much data is as bad as, having too little data – the network manager at the NMS has to make sense of all the received data, correlating alarms to isolate primary faults from secondary faults (faults that occur as side effects of the primary fault). Every network management application should be scalable, non – intrusive and robust. Many users of SNMP have felt the need for a more powerful and intelligent facility to aid the network manager.

The two basic questions in network fault management remain: How to model the network for fault management and what procedures to undertake for fault detection and identification. Two approaches identified in [1] include:

- 1) Taking a global (coarse) perspective of the network. This has led to the study of probabilistic models [8], [10].
- 2) Taking a detailed (fine) view of the network entities by modeling them as finite state machines (FSMs) [12].

In this research, we take a coarse perspective of the network, using Bayesian networks as our modeling and inference tool. The Bayesian network is suitable for problems involving uncertainty, requiring correlation of evidence and in which causality plays a role. R. Deng and colleagues [3] have applied Bayesian networks to linear lightwave networks. Hood and Ji [5] have also applied Bayesian networks to fault management, but from a different perspective: not to specify or identify faults, but to represent the network's health. However, we feel that to completely address the problem of fault management, we cannot avoid the identification of “known” faults. A good review of models and algorithms for network fault detection and identification is presented in [1].

3. Structural Overview

We adopt the fault identification process proposed by Bouloutas et al [9], which involves three steps:

- **Fault detection:** A fault is primarily characterized by malfunctions on individual network devices. For SNMP monitored devices, this means that several alarms will be generated as a result of some MIB variables crossing, either positively or negatively, some predefined threshold. These locally generated alarms can be correlated to determine the authenticity of the alarm and, as far as possible, the root cause of it.

- Fault Localization –The identification of the domain of the fault: the set of objects that might be the primary or secondary sources of the observed failure [10].
- Fault source identification - The correlation of the results of the fault detection step received from devices in the fault domain along side with other tests carried out on the network to reduce the uncertainty of the intelligent agent about the actual source and nature of the fault.

The first step is carried out by a subordinate (or service) agent resident on each monitored device, henceforth referred to as the S-agent. The second and third steps are carried out by the manager agent, put in charge of a collection of S-agents and hence forth referred to as the M-agent. Here we focus on the fault detection process on the S-Agent.

4. Fault Detection

It is commonplace to assume that network alarms are reliable indications of faults. This is not always true. There is enough redundant information in the MIB that can be correlated to confirm an alarm and to identify fault situations such as broadcast storms, network congestion, faulty interface cards and the like.

Effective modeling for network fault diagnosis should account: knowledge about the managed objects, and the propagation patterns faults tend to follow. We modeled the problem of network fault diagnosis using Bayesian Networks [6], [7], [11]. The Fault detection process proceeds in 3 steps: Situation Assessment, Observation Selection and Action Execution [2].

4.1 Situation Assessment

Situation assessment is the task of inferring an explanation for the observed alarms in the network. In probabilistic terms, this amounts to estimating the probability distribution of different faults that the S-agent can identify given the raised alarms. What follows is a brief explanation of the Bayesian network model and the algorithm for calculating this probability distribution.

A Bayesian network is a directed acyclic graph in which the nodes represent propositional variables, the arcs indicate the existence of a direct causal relationship between the connected nodes. Each parent-child cluster holds a conditional probability table quantifying the causal influence of parents on their common child. Figure 1 shows a sample Bayesian network that could be maintained by the S-agent. We identify five classes of nodes: **Fault nodes** (e.g. *Broadcast Storm*, *Network Congestion*) that represent the types of faults or fault conditions that this agent can identify, **Dynamic Nodes** (e.g. *ifInDiscards*) that represent MIB variables that change often such as counters, **Deterministic Nodes**, which are functions of other MIB variables, **Function Nodes** (e.g. *ifLoopBackTest- found in MIB-II*) which represent the

results of other tests on the network and *Static Nodes*, which represent MIB variables that rarely change.

The values of dynamic nodes and deterministic nodes are monitored at predefined intervals and alarms are raised by the monitoring component whenever these variables exceed their threshold. These alarms are passed as evidence (events) onto the Bayesian network. The nodes *Input buffer overflow* and *Output buffer overflow* can be modeled as fault nodes or deterministic nodes, depending on the network manager's outlook.

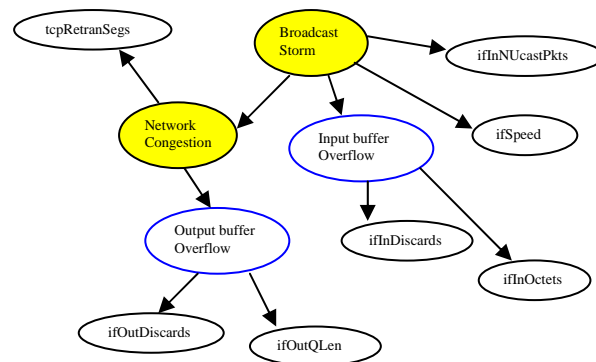


Fig. 1. Bayesian Network Model for a host s-agent

Each node in the graph can be seen as an autonomous object and the arcs as bi-directional inter-object communication channels. Each node stores a matrix, which is its conditional probability table, and a belief vector – which is the probability distribution over the possible states in which the node may exist. This belief vector gets updated as new evidence is received. On receipt of new evidence, the Bayesian network launches a sequence of message-passing to reach a new equilibrium state [6]. Each node can receive messages from its children and parents, update its belief vector and forward messages to its parents and children. To avoid double counting of evidence, each node is visited only once, ensuring that the network reaches equilibrium after a finite number of steps. Figure 2 depicts pictorially the situation assessment step.

Situation assessment however, may not necessarily yield an acceptable measure of certainty to warrant reporting a fault situation or taking any steps to correct the fault. The S-agent may need to query other *relevant* variables (static nodes and function nodes) to reduce its level of uncertainty.

Since function nodes hold the results of tests, it means these tests must be conducted. In this manner, other networking fault tests can be included in the reasoning cycle of the S-agent.

The S-agent has to decide how relevant a variable is to a particular problem, and this is defined as the degree of *informativeness* of the variable (or observation).

4.2 Observation Selection

The situation assessment phase results in a set of hypothesis faults as observed by the S-agent. In general, if we denote $\Phi = \{F_1, \dots, F_2\}$, the set of faults the agent can detect, the situation assessment returns a probability distribution over this set of faults. To measure the uncertainty of the S-agent, we use the entropy measure [13]. If we note $P(F_j)$ the probability that fault F_j is present, the entropy measure is defined as follows:

$$H(P) = -\sum_j P(F_j) \log P(F_j) \quad (1)$$

The entropy measure has many interesting properties. If all faults are equally likely, the S-agent has a very uncertain estimate of which object is faulty. The entropy is then maximal. As one fault candidate becomes likely, the entropy decreases, until desirably, reaching zero when one of the fault probabilities tends to one. The entropy measure quantifies the degree of uncertainty of the S-agent.

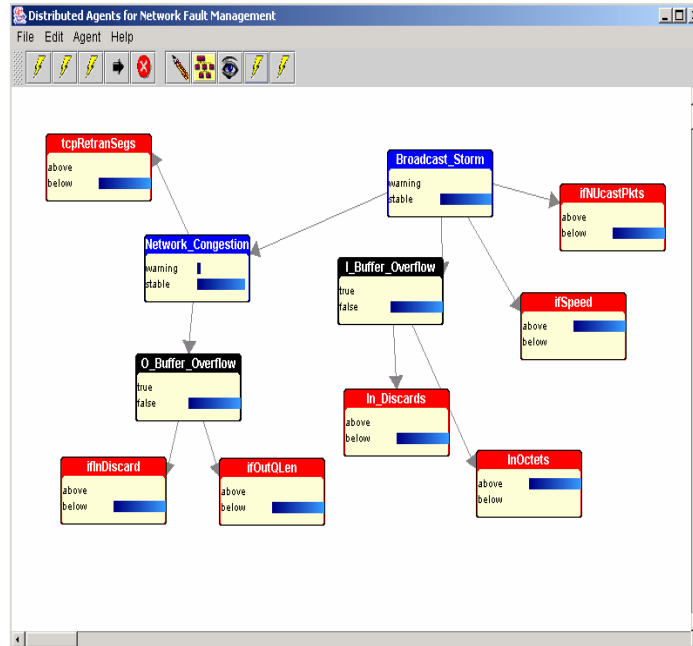


Figure 2. Situation Assessment of the S-agent

Let o be a possible observation variable the S-agent can consult to reduce its uncertainty, and assume it takes its value from the set $\{o_1, \dots, o_m\}$. If we observed the variable O , and found its value to be o_j , the entropy measure becomes:

$$H(P|o_i) = -\sum_j P(F_j|o_i) \log P(F_j|o_i) \quad (2)$$

Since we do not know the value of the variable O yet, the contribution of the observation O can be summarized by averaging the values of equation (2) by the probability distribution over the possible outcomes of O . That is:

$$H(P|O) = \sum_i H(P|o_i)P(o_i) = -\sum_i \sum_j P(F_j, o_i) \log P(F_j|o_i) \quad (3)$$

The potential contribution of an observation variable O can be calculated as follows:

$$I(P, O) = H(P) - H(P|O) \quad (4)$$

This quantity is non-negative. It becomes null when O does not contribute to any reduction in the uncertainty of the S-agent. Initially, the S-agent has to evaluate $I(P, O)$ for all possible observation variables and subsequently chose the observation O^* with the highest $I(P, O)$ and adds it to a set Θ . Next, all observations $\{o_1, \dots, o_k\}$ that are independent (using the d-separation criteria [11]) of Φ , given Θ are ignored as irrelevant. The process is repeated with the remaining observation variables.

The algorithm complexity depends mainly on the complexity of inference in Bayesian networks. Observation selection is performed in polynomial time if the network is singly-connected. For arbitrary complex topologies, the inference in Bayesian networks is NP-Hard [6].

4.3 Action Execution

Depending on the degree of automation of the network, and the nature of the identified fault, the S-agent may carry out local recovery actions. In the absence of any such ability, it will simply report its findings to the M-agent along with any other useful information.

5. Future Works

The values in the conditional probability table were subjectively defined. However, an important property of computer networks is the size of available data. This can be exploited to relieve the user from the burden of supplying exact probabilities. Retrieving conditional probabilities values from data sets is briefly explained in [11], and the normative theory of statistical learning is being extended to handle the case of incomplete data sets, and Bayesian networks with hidden variables [11, 14].

Generally, the predefined value of the threshold is no more than an estimation of the normal range within which the measured feature is believed to operate. There seems to be little objective insight on how to fix these thresholds. If the threshold is set too high, subtle changes will go unnoticed, and if too low, too many alarms will be generated. Fuzzy logic might be applied, providing fuzzy boundaries for threshold selection.

6. Conclusions

Although, the infrastructure for network management is largely standardized (SNMP and OSI CMIP), little has been done to define how the collected information from the managed objects is used to achieve fault, performance, accounting, configuration, and security management functions. The degree of automation provided by current systems is limited. This research presented a framework for both modeling and diagnosis of network faults. The Long term goal of this research is to develop a framework that will fully automate the fault identification and restoration process in computer networks. We addressed a small part of the whole exciting challenge of achieving autonomous and self-healing networks. This is a multidisciplinary task that will benefit from advances in methods for mobile agents, machine learning and models and algorithms for acting under uncertainty.

References

- [1] A.A. Lazar, W. Wang and R.H. Deng, “*Models and algorithms for network fault detection and identification: A review*,” Proc. Int’l Conf Communications, 1992.
- [2] Behrouz H. Far, “*Distributed Software Agents for Network Fault Management*,” University of Calgary. Internal Paper.
- [3] R.H. Deng, A.A. Lazar, W. Wang, “*A probabilistic approach to fault diagnosis in linear lightwave networks*,” IEEE Journal on selected areas in communications, vol. 11, No. 9, Dec. 1993.
- [4] W. Stallings. “*SNMP, SNMPv2, SNMPv3 and RMON 1 and 2*”, Addison-Wesley, 1999.
- [5] S.C. Hood and C. Ji, “*Intelligent Processing Agents for Network fault Detection*,” IEEE Internet Computing, 1997.
- [6] J. Pearl. “*Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*,” Revised Edition, Morgan Kaufmann Publishers. 1991.
- [7] F.V. Jensen, “*Bayesian Networks and decision Graphs*,” Springer-Verlag 1991.
- [8] R.A. Maxion, “*Anomaly detection for diagnosis*,” IEEE Int’l Conf’ on Fault Tolerant Computing, 1990.
- [9] A.T. Bouloutas, S. Calo, A. Finkel “*Alarm Correlation and Fault Identification in Communication Networks*,” IEEE Transactions on communications vol.42, No. 2/3/4, Feb/March/April 1994.
- [10] I. Katzela, M. Schwartz, “*Schemes for Fault Identification in communication Networks*” IEEE/ACM Transactions on networking, vol. 3, No. 6 Dec 1995.
- [11] N. J. Nilsson, “*Artificial Intelligence: A new synthesis*,” Morgan Kauffmann, 1998.
- [12] I. Rouvellou and G.W. Hart, “*Automatic Alarm Correlation for Fault Identification*,” IEEE Transactions 1995.
- [13] C. Shanon, “*A mathematical theory of communication*,” Bell System Technical Journal, 27: 379-623, 1948.
- [14] D. Heckerman, D. Geiger, D. Chickering. “*Learning Bayesian networks: the Combination of Knowledge and Statistical Data*,” Technical report, 1994.