CAPRI: A Common Architecture for Autonomous, Distributed Diagnosis of Internet Faults using Probabilistic Relational Models

George J. Lee
Massachusetts Institute of Technology
Computer Science and Artificial Intelligence Laboratory
Cambridge, MA 02139 USA
gjl@mit.edu

Abstract

Internet fault diagnosis today is slow, costly, and error-prone because it requires humans to run diagnostic tests and interpret their results. A fully autonomous self-diagnosing network could greatly improve diagnostic accuracy and efficiency, but such a network requires a common language for expressing diagnostic knowledge and data, and a protocol for distributed probabilistic diagnostic reasoning. In this paper I show how the Common Architecture for Probabilistic Reasoning in the Internet (CAPRI) can satisfy these requirements using probabilistic relational models (PRMs). Preliminary results indicate that CAPRI agents can diagnose HTTP proxy connection failures with over 80% accuracy using TCP failure data collected using an updated version of Planetseer[11].

1. Introduction

Internet fault diagnosis requires the application of diagnostic knowledge, performing diagnostic tests, and distributed diagnostic reasoning. When a user reports a network fault such as an IP reachability failure to a network administrator, the administrator diagnosing the fault uses their diagnostic knowledge to identify the data they need for diagnosis. Then they perform diagnostic tests to collect data about the fault, using tools ranging from simple ping and traceroute measurements to sophisticated SNMP network management consoles. Next they interpret the results of these tests to infer possible causes of failure. Due to the distributed administration of the Internet and dependencies among network components, frequently the administrator may also need to coordinate with users and administrators in other domains to diagnose the failure. Finally, the administrator identifies the root cause of failure and informs the user.

Today, humans perform all of the steps of diagnosis manually, but as the Internet grows in size and complexity this will become increasingly impractical. Though we have many tools for data collection[9, 11] and sophisticated diagnostic reasoning[4, 6], unfortunately we do not have a way to automate distributed Internet fault diagnosis using these tools. To address this problem we need a common architecture for distributed diagnosis of Internet faults using autonomous agents. This architecture must provide a common language for expressing diagnostic knowledge and data, and a protocol for distributed probabilistic reasoning.

In this paper I present the Common Architecture for Probabilistic Reasoning in the Internet (CAPRI), which uses probabilistic relational models (PRMs)[3] to support autonomous diagnosis of IP reachability and other failures. The primary research contribution of this paper is to show how to apply recent AI research in PRMs to provide a general yet expressive framework for autonomous distributed probabilistic diagnosis of Internet faults.

2. Requirements of a Diagnostic Architecture

A diagnostic architecture must provide a language for expressing diagnostic knowledge and data, and a protocol for distributed probabilistic reasoning.

2.1. A Language for Diagnostic Knowledge

Accurate diagnosis of network failures requires knowledge of the properties of both the classes of network components that might fail and the classes of tools available for testing these components. Network components include both processes such as TCP connections and devices such as Ethernet switches; diagnostic tests include both passive observations as well as active measurements. For example, to diagnose an IP path failure, one needs to know that an IP path failure results from a failure in one of the links along that path, and that a failed traceroute along an IP path can indicate a path failure. Today such knowledge mostly resides in the heads of network administrators; in order for autonomous agents to make use of such information we need a common language for expressing this diagnostic knowledge.

Furthermore, this language must be extensible because the set of network component and diagnostic test classes in the Internet is constantly expanding. In order for agents to diagnose new components and take advantage of new diagnostic tests, developers need the ability to specify new diagnostic knowledge and share this knowledge with existing agents.

Most previous architectures for exchanging diagnostic information in the Internet do not support the communication of such high-level diagnostic knowledge. Existing ontologies for representing information about network components such as the Common Information Model (CIM)[7] and architectures for exchanging network data such as Sophia[10] primarily deal with low-level knowledge of network components, and do not model the high-level problem-solving knowledge necessary for diagnosis. The high-level approach to diagnosis that I take in this paper fits with the concept of the Knowledge Plane[1], which aims to provide a common framework for exchanging and reasoning about network knowledge.

2.2. A Language for Diagnostic Data

Diagnostic agents also need a common language for expressing the diagnostic data produced by diagnostic tests and reasoning. A common language for expressing diagnostic data can unify the incompatible array of diagnostic tests and reasoning methods available today. Each class of diagnostic tests produces output in a different format, complicating automated reasoning. In addition, automated reasoning methods for diagnosis abound, ranging from Bayesian inference to case-based reasoning, each producing mutually unintelligible output. Having a common language for expressing the output of diagnostic tests and reasoning would enable autonomous diagnostic agents to combine information derived from multiple diagnostic tests and reasoners and exchange these results with other agents.

This language must provide enough generality to support the wide range of diagnostic tests and reasoning methods today while retaining enough expressiveness to permit agents to infer a diagnosis from data by applying diagnostic knowledge. Simply exchanging low-level measurements such as packet round trip times or TCP traces is neither general nor expressive; instead, we need to translate such information into a meaningful high-level diagnostic language that can express statements such as "The link between *A* and *B* is down," or "A ping along the path from *A* to *C* failed."

2.3. Distributed Probabilistic Reasoning

Diagnosis requires distributed probabilistic reasoning. Reasoning must be probabilistic because many diagnostic tests only indicate a probability of failure, and in many situations complete and accurate data is not available. Many researchers have studied probabilistic models for fault diagnosis[5, 2, 8, 4], but no common architecture for sharing diagnostic knowledge across administrative domains for distributed probabilistic reasoning exists. Probabilistic systems that use centralized reasoning are inadequate because diagnostic knowledge and data may be distributed across multiple administrative domains. For example, a network administrator may not have the data or knowledge necessary to diagnose failures in their upstream ISP.

In addition, an architecture for Internet-scale fault diagnosis must deal with the huge volume of diagnostic requests that may result from a serious high-impact failure that simultaneously affects a large number of users at a time when the network is already stressed. Thus a protocol for distributed reasoning must minimize the communication cost of both diagnostic tests and the message passing involved in reasoning.

3. CAPRI: Common Architecture for Probabilistic Reasoning in the Internet

In order to address the requirements above, I propose a Common Architecture for Probabilistic Reasoning in the Internet (CAPRI) that provides a framework for distributed probabilistic diagnostic reasoning. Unlike previous architectures for Internet fault diagnosis, CAPRI represents diagnostic knowledge and data using probabilistic relational models (PRMs)[3], combining the strengths of probabilistic Bayesian inference with the descriptive power of first-order logic. PRMs provide diagnostic agents with a language for expressing probabilistic diagnostic knowledge, while Bayesian networks enable agents to share probabilistic diagnostic data and perform distributed inference using belief propagation.

3.1. Expressing Diagnostic Data using Bayesian Networks

Bayesian networks compactly represent the conditional probability of related events and enable efficient inference based on available evidence[5]. A Bayesian network is a directed acyclic graph in which nodes represent variables, and edges from *parent* nodes to *children* nodes represent dependence relations. Each node X has a conditional probability table (CPT) $P(X|parents(X))$ that encodes the conditional probability of X given evidence about its parents.

In CAPRI, the variables in the Bayesian network represent component status and test results. A component status node in this Bayesian network represents the functional status of a component: `True` if functioning, `False` if malfunctioning. A test result node represents the output of a diagnostic test, `True` if successful, `False` otherwise. Edges in the Bayesian network represent dependencies among component status and diagnostic test results. The CPT for a component status node represents the probability that it is functioning given the status of its parent components. The CPT for a test node indicates the probability of a successful test given the status of its parent components.

Bayesian networks have several important features that make them especially suitable for reasoning about failures in the Internet. Firstly, Bayesian networks can model both deterministic and probabilistic dependencies among many types of Internet components and diagnostic tests. For example, an IP path functions if and only if the first hop link functions and the rest of the path functions. A ping along that path will always fail if the path has failed, but may fail 5% of the time even when the path is functioning. Individual links function 99% of the time. Figure 1 illustrates a Bayesian network that encodes this information for the path $A \rightarrow B \rightarrow C$. Using this network, a diagnostic agent can infer, for example, the probability that Link *B-C* has failed given evidence that Ping $A \rightarrow C$ has failed and Link *A-B* is functioning. To take into account evidence from active probing or changing network conditions, an agent can rebuild this Bayesian network from a PRM when new information becomes available (see Section 3.2).

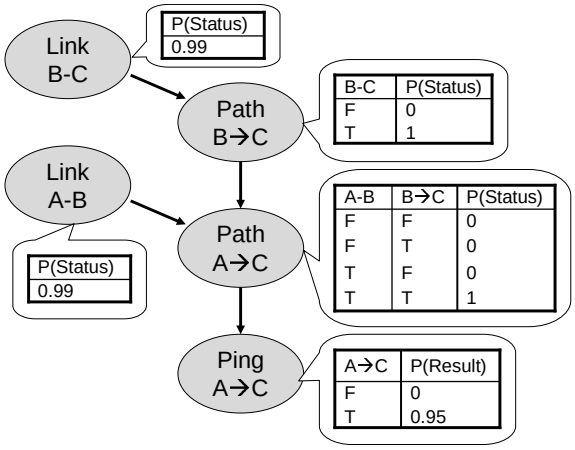

Figure 1. A Bayesian network for IP path diagnosis

Another advantage of Bayesian networks is that they provide an abstract high-level representation for diagnostic data suitable for reasoning. Representing diagnostic data in terms of variables, evidence, and dependencies rather than passing around low-level measurements such as packet traces allows an agent to reason about the causes and consequences of failures without any deep knowledge of the behavior and characteristics of components and diagnostic tests. This allows the architecture to support a wide range of components and diagnostic test classes, including new classes that researchers develop in the future.

The conditional independence assumptions of a Bayesian network facilitate distributed reasoning. For

example, an agent can infer that an IP path has failed if that agent has evidence that a link along that path has failed without knowing the cause of the link failure. This structure minimizes the number of other agents with which an agent needs to communicate to infer a diagnosis. Thus each agent can maintain only a local dependency model and perform distributed inference using a variation of loopy belief propagation[2], requesting updated beliefs from other agents when new evidence becomes available.

Probabilistic inference can greatly reduce the number of diagnostic tests required to infer the root cause of a failure compared to active probing methods such as Planetseer[11]. When high-impact failures occur and an agent receives many failure requests with the same root cause, Bayesian inference enables an agent to infer the root cause with high probability without additional tests. When an agent does not have enough information for diagnosis, an agent can determine which tests will provide the maximum amount of diagnostic information and perform only those tests[8].

Probabilistic inference also enables agents to provide diagnosis even when they cannot obtain accurate data due to failures or lack of information. For example, if the agent responsible for diagnosing IP connectivity failures in an Internet autonomous system (AS) *X* is unreachable, another agent can still infer the most probable explanation for a failure in AS *X* based on historical IP link failure probabilities.

3.2. Expressing Diagnostic Knowledge using PRMs

Probabilistic Relational Models (PRMs) provide a language for expressing the shared diagnostic knowledge of component and test classes in a way that supports automated diagnostic inference while enabling the definition of new component and test classes in terms of existing classes. A PRM defines classes and properties of individuals belonging to each class, together with parent relationships and CPTs for those properties. Figure 2 illustrates the PRM representing the diagnostic knowledge for IP path failures. The boxes in the figure represent classes, the entries within a box represent properties of individuals of that class, dotted lines represent foreign key relationships relating individuals of two classes, and arrows represent

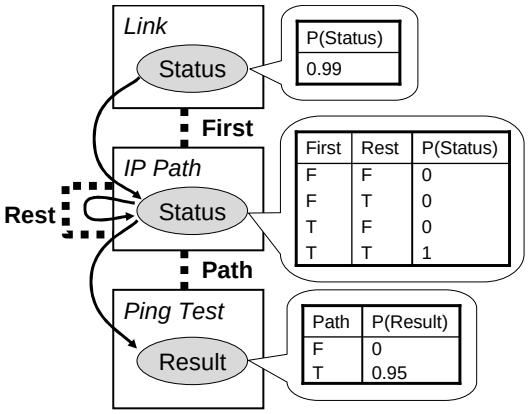

Figure 2. A PRM for IP path diagnosis

parent relationships. Using this PRM, an agent can construct a Bayesian network such as the one in Figure 1 using runtime dependency analysis and local information. For clarity Figure 2 only shows part of the PRM for IP path diagnosis; other factors that a diagnostic agent might take into consideration include the properties of links, the time a diagnostic test is performed, the cost of diagnostic tests, the administrators of network components, and so on.

Diagnostic knowledge may come either from human experts or from Bayesian learning. For many component classes such as IP paths, CPTs simply deterministically encode truth tables for AND or OR. In such cases application developers or other experts can easily specify the CPT. Even if the exact conditional probabilities are unknown, however, agents can learn them collectively using Bayesian learning[3].

For extensibility to support new classes, CAPRI allows subclassing where each subclass may have a different CPT. For instance, rather than modeling all individuals belonging to the *Link* class as having the same probability of failure, for increased accuracy we can distinguish between two subclasses *Wired Link* and *Wireless Link*. This captures the diagnostic knowledge that individuals of the *Wireless Link* class have a much greater probability of failure than *Wired Link* individuals. Subclassing provides a mechanism for introducing new classes of components and tests while retaining compatibility with existing diagnostic agents.

In addition, defining component classes in terms of a PRM enables agents to use previously learned knowledge to diagnose new classes of components.

For example, if someone develops a new application class *NewApp* that depends on two *HTTP Connections*, then given diagnostic knowledge about the dependencies of *NewApp*, any agent that can diagnose *HTTP Connections* can diagnose *NewApp* failures. Agents can share knowledge of new classes to create a distributed, extensible ontology using the Web Ontology Language (OWL)[1].

4. Experimental Evaluation

To evaluate the effectiveness of probabilistic diagnosis for real-world Internet connectivity failures, I trained and tested a Bayesian network for diagnosis on an artificial set of HTTP proxy connections using data on 28.3 million TCP connections observed over 196 CoDeeN nodes using an updated version of Planetseer[11]. I find that when an HTTP proxy connection failure occurs, knowing only the AS numbers of the source, proxy, and destination a CAPRI agent can determine which TCP connection has failed (either user/proxy or proxy/server) with over 80% accuracy, compared to 62% accuracy for a deterministic diagnostic agent. In addition, the probability of TCP failures between ASes stays relatively constant over time; the accuracy of diagnosis remains greater than 76% after nine hours, indicating that the TCP failure probabilities learned in this experiment are useful for the diagnosis of future failures as well. These initial results show how an architecture for probabilistic reasoning using Bayesian networks enables agents to accurately learn about and diagnose failures in the Internet.

5. Conclusion and Future Work

In this paper I show how CAPRI can support autonomous fault diagnosis by using PRMs to provide a common language for expressing diagnostic knowledge and data and provide a protocol for distributed probabilistic diagnostic reasoning. I am currently conducting additional experiments to quantify the accuracy and communication cost of distributed TCP path diagnosis in the Internet using CAPRI. In future work I plan to investigate the use of dynamic probabilistic relational models (DPRMs) for modeling temporal dependencies among components and diagnostic tests.

[1] http://www.w3.org/TR/owl-ref/

Acknowledgments

I am grateful to Vivek Pai and Lindsey Poole for providing access to TCP connection data from the updated version of Planetseer.

References

[1] D. D. Clark, C. Partridge, J. C. Ramming, and J. T. Wroclawski. A knowledge plane for the internet. In *Proceedings of SIGCOMM '03*, 2003.

[2] C. Crick and A. Pfeffer. Loopy belief propagation as a basis for communication in sensor networks. In *Proceedings of the 19th Annual Conference on Uncertainty in Artificial Intelligence (UAI-03)*, 2003.

[3] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, Stockholm, Sweden, August 1999.

[4] S. Kandula, D. Katabi, and J.-P. Vasseur. Shrink: A Tool for Failure Diagnosis in IP Networks. In *ACM SIGCOMM Workshop on mining network data (MineNet-05)*, Philadelphia, PA, August 2005.

[5] U. Lerner, R. Parr, D. Koller, and G. Biswas. Bayesian fault detection and diagnosis in dynamic systems. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00)*, pages 531–537, Austin, Texas, August 2000.

[6] V. N. Padmanabhan, S. Ramabhadran, and J. Padhye. Netprofiler: Profiling wide-area networks using peer cooperation. In *Proceedings of the Fourth International Workshop on Peer-to-Peer Systems (IPTPS)*, February 2005.

[7] S. Quirolgico, P. Assis, A. Westerinen, M. Baskey, and E. Stokes. Toward a formal common information model ontology. In *WISE 2004 Workshops*, 2004.

[8] I. Rish, M. Brodie, N. Odintsova, S. Ma, and G. Grabarnik. Real-time problem determination in distributed systems using active probing. In *Proceedings of Network Operations and Management Symposium (NOMS 2004)*, 2004.

[9] N. Spring, D. Wetherall, and T. Anderson. Scriptroute: A public internet measurement facility. In *Proceedings of USENIX Symposium on Internet Technologies and Systems (USITS)*, 2003.

[10] M. Wawrzoniak, L. Peterson, and T. Roscoe. Sophia: an information plane for networked systems. *SIGCOMM Comput. Commun. Rev.*, 34(1):15–20, 2004.

[11] M. Zhang, C. Zhang, V. Pai, L. Peterson, and R. Wang. Planetseer: Internet path failure monitoring and characterization in wide-area services. In *Proceedings of Sixth Symposium on Operating Systems Design and Implementation (OSDI '04)*, 2004.