Diagnosis of TCP Overlay Connection Failures using Bayesian Networks

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

Lindsey Poole
Department of Computer Science
Princeton University
Princeton, NJ 08544 USA
lpoole@cs.princeton.edu

ABSTRACT

When failures occur in Internet overlay connections today, it is difficult for users to determine the root cause of failure. An overlay connection may require TCP connections between a series of overlay nodes to succeed, but accurately determining which of these connections has failed is difficult for users without access to the internal workings of the overlay. Diagnosis using active probing is costly and may be inaccurate if probe packets are filtered or blocked. To address this problem, we develop a passive diagnosis approach that infers the most likely cause of failure using a Bayesian network modeling the conditional probability of TCP failures given the IP addresses of the hosts along the overlay path. We collect TCP failure data for 28.3 million TCP connections using data from the new Planetseer overlay monitoring system and train a Bayesian network for the diagnosis of overlay connection failures. We evaluate the accuracy of diagnosis using this Bayesian network on a set of overlay connections generated from observations of CoDeeN traffic patterns and find that our approach can accurately diagnose failures.

Keywords

Bayesian networks, fault diagnosis, passive diagnosis, Planetseer, TCP overlay path diagnosis

1. INTRODUCTION

When failures occur in Internet overlay connections today, it is difficult for users to determine the root cause of failure. The proliferation of TCP overlays such as content distribution networks and HTTP proxies means that frequently network communication requires a series of TCP connections between overlay nodes to succeed. For example, an HTTP request using the CoDeeN[9] content distribution network first requires a TCP connection to a CoDeeN node and then a connection from a CoDeeN node to a server or another CoDeeN node. A failure in any one of the TCP connections along the overlay path causes the user's HTTP request to fail. If the user knows which TCP connection failed, then they can take appropriate action to repair or circumvent the failure. For instance, if they know that the connection from the proxy to the server failed, then they complain to the web server administrator. On the other hand, if the user/proxy connection fails, perhaps they can try connecting to the proxy using a different ISP. If multiple overlay paths exist between the source and destination, nodes and applications may also use this type of diagnostic information to automatically recover or route around failures[1].

Unfortunately, accurately determining which TCP connection in an overlay connection has failed is difficult for end users, who typically do not have access to the internal workings of the overlay. Commercial overlay networks such as Akamai typically do not reveal details of connection failures to users, and the diagnostic tools available to users today are frequently inadequate. Active probing techniques such as tulip[7] and Planetseer[11] frequently cannot provide accurate information due to firewalls and packet filtering. Furthermore, active probing can be costly both in terms of network resources and time, and cannot diagnose the many transient TCP failures that begin and end before one can complete a probe[11]. Additionally, one must take care when using active probing for diagnosis because they may concentrate network traffic at points of failure and trigger intrusion detection systems.

Instead, in our research we consider a passive approach to diagnosis in which intelligent diagnostic agents use probabilistic inference to determine the root cause of failure. The reliability of IP links in the Internet varies widely and hence we expect the probability of TCP failure to differ between different sets of hosts. Diagnostic agents in the Internet learn the probability of such failures for different regions in the Internet based on observations of TCP traffic. When users or network administrators detect network failures, they request diagnosis from such diagnostic agents. Agents then use information about the relative probability of failure of the TCP connections that make up an overlay connection to identify the most likely cause of failure when an overlay connection occurs without conducting any additional probes. In addition, diagnostic agents can also use this Bayesian network to predict the probability of overlay and TCP connection failure given information about the path of an overlay connection.

We collect data on TCP failure probabilities in order to determine whether this data enables diagnostic agents data to accurately diagnose overlay failures in the Internet. To learn the probability of failure for TCP connections between different points in the network, we observe TCP traffic on the content distribution network CoDeeN using an updated version of Planetseer[11]. Next we construct a Bayesian network for diagnosis using these probabilities. We then use Bayesian inference to infer the most probable cause of failure for TCP-based applications.

To evaluate the effectiveness of this approach, we test this Bayesian network on an artificial set of overlay connections based on the

SIGCOMM'06 Workshops September 11-15, 2006, Pisa, Italy.
Copyright 2006 ACM 1-59593-417-0/06/0009 ...$5.00.

traffic observed on CoDeeN. We find that when a failure occurs, knowing only the AS numbers of the source, proxy, and destination, we can determine which TCP connection has failed with over 80% probability. In addition, the probability of failure between ASes stays relatively constant over time, and data learned can be accurately used for diagnosis for many hours into the future. This suggests that the TCP failure probabilities we learn may be useful in the diagnosis of future failures as well.

The contribution of this research is to show how inter-AS TCP failure probabilities can be used for probabilistic diagnosis of failures in overlay networks such as CoDeeN using Bayesian inference. We also demonstrate a variety of clustering methods to address the problem of dataset sparsity for learning TCP failure probabilities. In this paper we evaluate our system on CoDeeN overlay connections, but our Bayesian model generalizes to the diagnosis of other TCP-based applications as well.

2. RELATED WORK

There has been previous work in passive diagnosis of failures in the Internet. Padmanabhan, Ramabhadran, and Padhye developed Netprofiler, which collects network measurements from a set of end hosts and attempts to identify cause of failure by examining the shared dependencies among hosts that experience failures[8]. They show that this approach can provide information useful for diagnosis, but their paper only provides some preliminary results and do not provide details of how their system might diagnose real-world failures in practice.

Shrink probabilistically diagnoses IP link failures based on the observed status of IP links that share resources[4]. Similarly in our work we diagnose failures in overlay connections where an overlay depends on several underlying TCP connections which may share IP hops. Shrink assumes that one can accurately determine the status of all IP links at any point in time. This allows one to identify the shared cause of failure of the failed IP links. Theoretically, we can also use this approach to diagnose overlay failures. That is, we can identify the TCP connections that share common IP hops and observe which overlay connections have failed at any point in time to identify the failed TCP connections.

Unfortunately, in real-world diagnosis of TCP connections many of the assumptions made by systems such as Shrink do not hold for the following reasons.

1. The status of overlay connections may change rapidly, making it difficult to correlate failures in different overlay connections over time.

2. In order to construct a Bayesian network that accurately models the IP hops shared among different TCP connections we need an accurate IP level map of the Internet. As the Skitter[1] project demonstrates, accurately constructing such a map is difficult because routes may change and frequently tools such as traceroute do not provide accurate information.

3. Determining the status of an inactive overlay connection or a TCP connection is costly and takes time because it requires an active probe such as a ping, traceroute, or HTTP connection. Furthermore such probes are frequently inaccurate because of the prevalence of packet filtering, network address translation (NAT), and firewalls in the Internet[3].

4. TCP and IP failures are frequently so transient that by the time one can test the status of a link, the failure no longer exists [11].

Therefore in this paper we present an alternative passive diagnosis approach that does not require simultaneously knowing the status of all overlay connections. Instead, we cluster TCP failures based on the Internet autonomous systems (ASes) of their endpoints and use information about the distribution of TCP failures to infer the cause of failure. An agent first learns a probabilistic model of failures based on a training set of observed TCP connections, and then it uses this model to diagnose future failures when it does not know the connection status.

Other researchers have developed methods for diagnosing specific TCP-based applications. Ward, et al. infer the presence of TCP performance failures based on the rate of requests processed at an HTTP proxy server and TCP connection state [10]. Unlike such specialized diagnostic systems, our Bayesian approach to diagnosis can generalize to other applications that rely on TCP connections.

Most previous research in probabilistic diagnosis of Internet failures evaluate their work on simulated failures. Steinder and Sethi model network faults using a bipartite causality graph in which the failure of individual links cause the failure of end-to-end connectivity, and then perform fault localization using a belief network[6]. In contrast, in our research we evaluate our approach on real-world TCP failures using actual data collected on the Internet.

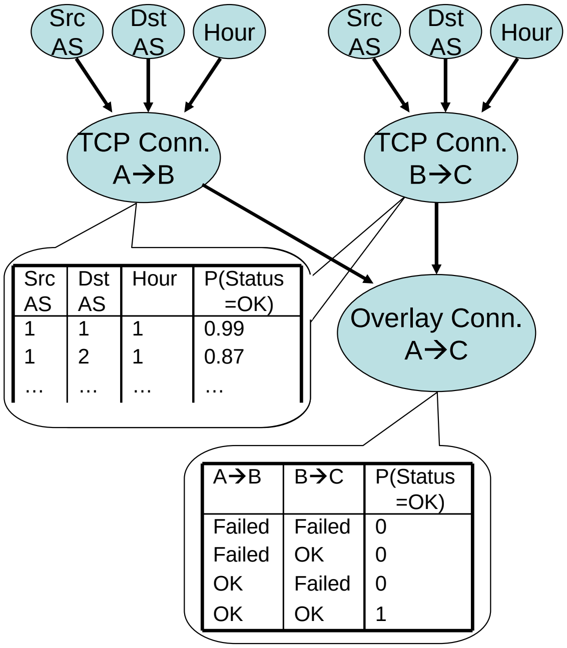
3. DIAGNOSING OVERLAY CONNECTION FAILURES

In this paper we consider the diagnosis of overlay networks in which an overlay network connection requires a series of TCP connections between overlay nodes between the source and destination hosts. For example, Akamai is a content distribution network in which retrieving a resource from a web server may require communication among multiple Akamai nodes along multiple TCP connections. Another example is the content distribution network CoDeeN on Planetlab, in which overlay nodes act as HTTP proxies. An request on CoDeeN[9] first requires a TCP connection to a CoDeeN node and then a connection from a CoDeeN node to server or another CoDeeN node. A failure in any one of these TCP connections causes the user's HTTP connection to fail. The challenge is to determine which of these TCP connections has failed.

Sometimes users can determine whether a failure has occurred along the first TCP connection along the overlay path using information provided by their local TCP stack, but if a failure occurs beyond the first connection users cannot tell where a failure occurs without cooperation from the overlay. Depending on the type of overlay, users may have different amounts of information about the overlay path. For example, in an HTTP proxy connection, users know that the proxy is the first hop along the path and that if the connection is not cached, the web server is the last hop along the path.

As a first step, in our research we examine a special case of diagnosis in order to gain insight into how well our approach might generalize to other types of diagnosis. The question we wish to answer is, if a two hop overlay connection fails due to a TCP failure, which TCP connection failed? In this paper we define a TCP failure as three consecutive TCP retransmits without a response. We assume that the diagnostic agent only knows that the overlay connection has failed and does not know which of the TCP connections has failed. We want to answer this question knowing only the IP addresses of the source, IP address of the first hop overlay node, and the IP address of the ultimate overlay destination host. Our model for probabilistic diagnosis generalizes to overlay connections with any number of hops, but as a starting point in this paper we only consider overlay connections with two hops.

[1] http://www.caida.org/tools/measurement/skitter/



Src AS	Dst AS	Hour	P(Status =OK)
1	1	1	0.99
1	2	1	0.87
…	…	…	…

A→B	B→C	P(Status =OK)
Failed	Failed	0
Failed	OK	0
OK	Failed	0
OK	OK	1

Figure 1: A Bayesian network for TCP overlay path diagnosis

3.1 Probabilistic Diagnosis

The reliability of IP links in the Internet varies widely and hence we expect the probability of TCP failure to differ between different sets of hosts. Thus if we have knowledge of the relative probability of failure of the TCP connections that make up an overlay connection, we can then infer the most likely cause of failure when an overlay connection occurs without conducting any additional probes. In this paper we show we can use Bayesian networks both to learn a model of TCP failures and to perform diagnosis.

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.

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 HTTP proxy connection functions if and only if the user/proxy TCP connection functions and the proxy/provider TCP connection functions. The probability that a TCP connection functions depends on the source and destination IP addresses and the time of the connection. To improve accuracy, we cluster IP addresses by AS and connection time by hour (see section 3.2). Figure 1 illustrates a Bayesian network that encodes the conditional probabilities for diagnosing an overlay connection from A to B to C. To diagnose an overlay connection failure from A to C, one can use this Bayesian network to infer the most probable status of the underlying TCP connections from A to B and B to C given information about the AS numbers and hour the connections were made.

The variables in the Bayesian network represent the functional status of TCP connections and overlay connections. A node in this Bayesian network represents the functional status of a connection: `OK` if functioning, `Failed` if malfunctioning. Malfunctioning means that a connection failure occurs along the path, functioning means that no connection failure occurs. Edges in the Bayesian network represent dependencies among connections. The CPT for an overlay connection node represents the probability that it is functioning given the status of its underlying TCP paths. The CPT for a TCP path represents the probability that the TCP path functions given information about the path. In our Bayesian network we assume that the conditional probability of a TCP connection failure depends only on the source and destination IP addresses and the time of failure for each hop of the overlay, and not on which hop of the overlay connection it is (user/proxy or proxy/server). We represent this by using parameter tying in this Bayesian network so that both TCP paths share the same CPT. We also assume that a diagnostic agent can identify the intermediate hops in the overlay connection, either through active probing or because it has knowledge of the overlay topology.

An advantage of modeling network components in terms of Bayesian networks is that a Bayesian network provides 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. In addition, the conditional independence assumptions of Bayesian inference reduce the amount of data a diagnostic agent needs to consider for diagnosis.

3.2 Clustering

To perform diagnosis using this Bayesian network, we need to learn the conditional probability of failure of a TCP connection given the properties of a connection. Learning the conditional probability of failure for each pair of IP addresses is impractical because it is infeasible to store the probability of failure for the 2^{64} combinations of source and destination IP addresses. More importantly, for each pair of IP addresses we only have a limited amount of data with which to train the Bayesian network. For more effective diagnosis, diagnostic agents need a way to diagnose failures involving IP addresses it has not previously observed.

Therefore to reduce the size of the conditional probability tables and to improve the accuracy of the learned probabilities, we cluster together IP addresses in a way that facilitates learning and diagnosis. Our hypothesis is that TCP connections that share many IP links with one another will have similar probabilities of failure. Thus two TCP connections with topologically nearby sources and nearby destinations will likely have similar failure probabilities. Therefore we clustered source and destination IP addresses in three ways: by the first eight bits of the IP address, the AS number, and by country.

We also cluster TCP connections based on time. We hypothesize that the probability of failure changes over multiple time scales. For instance, if an IP routing change occurs, the probability of failure for affected TCP connections may change from low to high and back to low within a few minutes. On the other hand, the average rate of routing failure over several days may remain relatively constant. We show how different methods for clustering affect the accuracy of diagnosis in section 5.

4. COLLECTING TCP FAILURE DATA

It is difficult to obtain accurate information about the distribution of TCP failures in the Internet because failed connections make up only a small percentage of overall TCP traffic and the number of possible source and destination IP addresses is enormous. To collect accurate failure probabilities, we need a way to observe the status of large quantities of TCP connections from many different source and destination hosts.

In order to obtain such data, we used an updated version of Planetseer to collect data on TCP connection failures. The new Planetseer monitors TCP connections in the CoDeeN content distribution network and provides notifications when TCP sessions begin, end, and when TCP failures occur. Planetseer runs on over 320 Planetlab[2] nodes distributed around the world. We used Planetseer to monitor all the TCP connections made by 196 CoDeeN nodes. We observed 28.3 million TCP connections and 249,000 TCP failures over a ten hour period. We observed TCP connections to approximately 17,000 distinct IP addresses per hour on average. In our dataset, we observed TCP connections to hosts in 2116 unique Internet autonomous systems.

CoDeeN overlay nodes act as HTTP proxies and establish TCP connections with web clients, web servers, and other CoDeeN nodes. In a typical CoDeeN session, a user initiates a TCP connection with the CoDeeN proxy, the proxy connects to a web server and retrieves the requested resource, and finally the proxy sends the requested data back to the user. Note that many requests are cached, and so the destination of the second hop in the overlay is a CoDeeN node and not the web server specified in the HTTP request. We found that 0.28% of user/proxy connections and 0.65% of proxy/server connections experienced TCP failures. Since Planetseer monitors TCP connections from the vantage point of the proxy, we cannot detect those TCP failures in which a user is unable to establish a TCP connection to the proxy. Therefore the lower percentage of user/proxy failures may be partly explained by the fact that all failures between the proxy and user occur after the user successfully establishes a TCP connection to the proxy.

We believe that the failure probabilities learned through Planetseer are representative of typical TCP connections in the Internet. CoDeeN nodes operate as HTTP proxies, so the pattern of TCP connections resembles typical web traffic. Though caching at CoDeeN nodes reduces the number of connections to web servers we observe, we believe that the average failure probability to web servers we observe using Planetseer reflects typical failure rates for HTTP related TCP connections. We are currently examining other types of overlay connections to determine how well this TCP data generalizes for the diagnosis of other overlays.

We learn the conditional probability table for TCP connection failure using the data collected from Planetseer. We cluster source and destination IP addresses by AS using the Oregon Route Views BGP tables[2].

5. EVALUATION

Our hypothesis is that Bayesian inference using the conditional probability of failure for TCP connections given the AS numbers of the source and destination can accurately diagnose failures in overlay connections. In order to test this hypothesis, we constructed a Bayesian network using the probabilities learned from Planetseer and used it to diagnose failures in CoDeeN connections.

We wanted to answer the following questions in our experiments:

1. Which clustering method produces the most accurate diagnosis: AS, IP/8 prefix, or country? We expect that clustering based on AS will produce the most accurate results since it is most closely correlated with the Internet routing topology.

2. How does diagnostic accuracy change as we increase the time interval over which we cluster TCP connections? We expect that as the clustering interval increases, accuracy will increase at first, but then decrease as the learned probabilities less accurately reflect the probabilities of new failures.

3. How does the age of the training set affect diagnostic accuracy? We expect that as the distribution of TCP failures in the Internet changes over time, diagnostic accuracy will also decrease.

5.1 Experimental Setup

We train a Bayesian network using the Bayes Net Toolbox (BNT) for Matlab[3]. In order to diagnose TCP connections between regions we did not observe in the training set, we initialize the prior probabilities of failure according to a uniform Dirichlet distribution, which is equivalent to adding an additional element to the training set for each combination of source cluster, destination cluster, and connection status. We test this Bayesian network on an artificial dataset generated based on the distribution of TCP connections observed on Planetseer. Since Planetseer does not provide information about which TCP connections are associated with each CoDeeN request, we construct a dataset based on the TCP connections we observed. First we identify user/proxy, proxy/proxy, and proxy/server connections based on IP address and port number. Then for each proxy, we count the number of TCP connections to each server and to each proxy. We assume that the number of cached requests equals the number of user/proxy connections minus the number of proxy/server and proxy/proxy connections. We assign each user/proxy TCP connection a corresponding proxy/provider connection, where the provider may either be a web server (if the resource is not cached), another proxy (if the resource is cached at another proxy), or the same proxy (if the resource is cached locally). We make these provider assignments according to the observed distribution of proxy/server and proxy/proxy connections. Of the 19,700 failures in this dataset, approximately 82% of requests are cached locally, 7.9% are cached at other CoDeeN nodes, and 10.6% are uncached.

For each CoDeeN request failure our Bayesian network makes two diagnoses: one for the status of the user/proxy connection, and one for the status of the proxy/provider connection. We measure accuracy in terms of the fraction of correct diagnoses. To evaluate the accuracy of diagnosis, we compute the most probable explanation for a TCP failure given evidence that the overlay connection has failed and the AS numbers of the source, proxy, and destination, and then compare this diagnosis with the actual status of the source/proxy and proxy/provider connections. In our experiments we perform diagnosis without evidence about whether a resource is cached at a proxy.

Of the CoDeeN requests that failed in the first hour of our dataset, we found that 62% failed at the user/proxy connection, 31% failed at the proxy/server connection, and 7% failed at a the proxy/proxy connection. Therefore knowing only the overall distribution of TCP failures between users and servers, without using information about the IP addresses of the user, proxy, and server, one could diagnose failures with 62% accuracy by diagnosing every failure as a user/proxy failure. In our experiments we wish to determine if our Bayesian approach to diagnosis can achieve significantly better accuracy.

In order to properly compute the accuracy of diagnosis, we separated the set of TCP connections with which we trained the Bayesian network from the set of TCP connections associated with the failed

[2] http://www.routeviews.org/

[3] http://www.cs.ubc.ca/ murphyk/Software/BNT

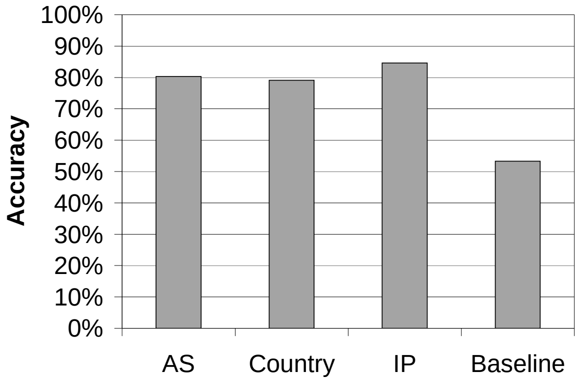


Figure 2: Clustering Method Comparison

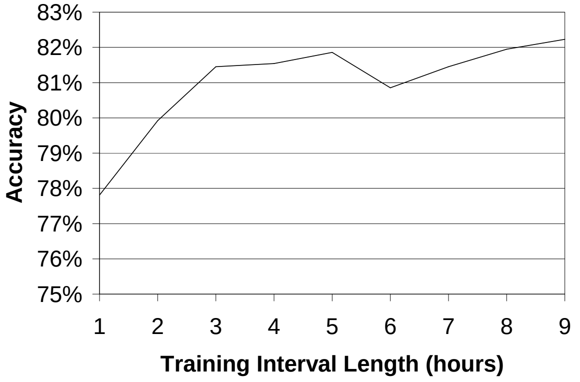


Figure 3: Accuracy vs. Training Interval Length

overlay connections under diagnosis. We collected ten hours of TCP connection data from Planetseer. In our initial experiments we choose to learn the average probability of failure over one hour because we find that clustering over shorter time scales does not provide enough data for accurate diagnosis.

5.2 Experimental Results

First we compare the accuracy of three IP clustering methods: by Internet autonomous system number (AS), by the first eight bits of the IP address (IP), and by the country in which a host resides (Country). We determine the country of a host using the hostip.info database[4], which maps the first 24 bits of an IP address to a country using location information contributed by Internet users. We train three Bayesian networks corresponding to the three clustering methods using data from hour 1. Then we test these Bayesian networks on the proxy connection failures constructed using data from hours 2–10 and averaged the results. We use a junction tree inference engine to compute the most likely status for each TCP connection and compare the inferred status with the actual status from the data. Since the Bayesian network we use for inference has no cycles, we can perform Bayesian learning and junction tree inference rapidly; in our experiments, inference for a single connection requires approximately 5 ms.

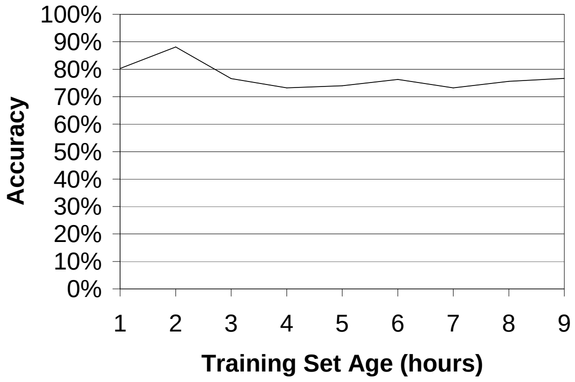


Figure 4: Accuracy vs. Training Set Age

Figure 2 compares the diagnostic accuracy of these three clustering approaches. We define accuracy as the fraction of correct status inferences. As a baseline, we also plot the accuracy of simply guessing that every failure is due to a user/proxy connection failure. Figure 2 shows that all three clustering methods provide similar degrees of accuracy. Our hypothesis was that clustering based on AS would produce the most accurate results, but our experiments show that clustering based on the first 8 bits of the IP address yields higher accuracy for short time intervals. This may be because one hour is not enough time to accurately learn inter-AS TCP failure probabilities, or due to inaccuracies in the Route Views BGP table.

Next we computed the accuracy of diagnosis as we increase the time interval over which we cluster TCP connections. If the interval over which we train is too short, then we will not have enough data to accurately learn failure probabilities. If the interval is too long, then it may not accurately reflect changing network conditions. We train a Bayesian network using AS clustering on *x* hours before hour 10 for values of *x* from 1 to 9. We then test each Bayesian network on the data from hour 10. Figure 3 shows how accuracy changes as the training time interval changes. This plot shows that accuracy increases as the clustering time interval increases, suggesting that the training value of incorporating additional data outweighs the inaccuracy introduced by using older data.

Finally, we compute the accuracy of diagnosis as we increase the age of the data on which we trained the Bayesian network. We train a Bayesian network using AS clustering on data from hour 1 and test it on overlay failures observed during each of the hours from 2 to 10. Figure 4 plots the accuracy of diagnosis over time. Average accuracy changes over time because the distribution of failures we observe using Planetseer varies from hour to hour, but overall diagnostic accuracy diminishes only slightly after nine hours, suggesting that the distribution of TCP failure probabilities remains relatively stationary over time.

We also compare the false positive and false negative rates for each clustering method. The false positive rate is the fraction of functioning connections that are incorrectly diagnosed as having failed, while the false negative rate is the fraction of failed connections that are incorrectly diagnosed as functioning. Table 1 lists the false positive and false negative rates for each clustering method.

5.3 Analysis

These experiments show that we can diagnose overlay connection failures knowing only the AS numbers of its TCP endpoints.

[4] http://www.hostip.info/

	AS	Country	IP	Baseline
user/proxy false pos.	0.174	0.358	0.426	1.000
user/proxy false neg.	0.219	0.050	0.060	0.000
proxy/server false pos.	0.219	0.101	0.265	0.000
proxy/server false neg.	0.171	0.128	0.100	1.000

Table 1: Diagnosis error rates by type

One reason our approach to diagnosis works is due to the heavy-tailed distribution of TCP connection failure probability. The majority of TCP failures occur among a small number of AS pairs. Therefore most CoDeeN connection failures involve one TCP connection with low failure probability and another TCP connection with high failure probability, so probabilistic inference produces the correct diagnosis. For example, we find that TCP connections from hosts in China to hosts in the USA tend to have a much higher probability of failure than connections within the USA. If an CoDeeN user in China accesses a proxy in the USA to retrieve content from a web server in the USA and experiences a failure, then it is very likely that the failure occurred on the connection between the user and the CoDeeN node. If the probability of failure for every pair of ASes were equal, then our probabilistic approach to diagnosis would not work as well.

Another interesting result is that the accuracy of diagnosis diminishes relatively slowly over time, implying that the distribution of TCP failures in the Internet stays relatively stationary over time. This suggests that diagnostic agents can perform accurate diagnosis using inter-AS TCP failure probabilities without having to constantly collect the latest TCP failure data.

6. CONCLUSION AND FUTURE WORK

Our initial experimental results indicate that our passive probabilistic approach to diagnosing TCP overlay connection failures can provide useful diagnostic information. In this paper we show that Bayesian inference provides a useful framework for diagnosing two hop overlay connection failures on CoDeeN, but our approach can generalize to the diagnosis of other overlay connection failures as well. We view our approach to diagnosing TCP overlay connection failures as just one example of a more general probabilistic approach for Internet fault diagnosis. In this paper we show how to use inter-AS TCP failure probabilities to diagnose failures in overlay networks, but the technique we used to diagnose failures in CoDeeN can be extended to the diagnosis of other overlays as well. We can apply the knowledge we learned from Planetseer to diagnose other classes of network components and applications by adding new nodes and edges to the Bayesian network we use for diagnosis.

In this paper we only considered diagnosis without using any additional evidence about a failure. Typically, however, when failures occur users may already know the status of certain network components and can perform diagnostic probes to collect additional evidence for diagnosing failures. We can improve the accuracy of our approach by adding variables and edges to the Bayesian network to take into account this information. For instance, if we know the IP paths that TCP connections traverse, we can incorporate evidence of IP link failures into the Bayesian network. We intend to explore how agents can incorporate such additional evidence into a Bayesian network to improve diagnostic accuracy.

In future work we will also examine more accurate models for Internet fault diagnosis that take into account failures at both short and long time scales. In this paper we only evaluated our algorithm on ten hours of data from Planetseer; we would like to conduct additional experiments to more accurately determine the effectiveness of diagnosis using data from other time periods as well. In addition we would like to explore other clustering methods, including dynamically choosing the prefix length on which to cluster based on how much data an agent has about TCP connections to a particular IP range.

Finally, though our paper describes a centralized diagnosis approach, this approach can easily be adapted for distributed diagnosis. Knowledge of the overlay topology and the conditional probabilities in the CPTs can be distributed among multiple agents in the Internet, allowing different agents to collect failure data from different points in the network. We are currently developing such a distributed system for the diagnosis of TCP application failures in the Internet.

7. REFERENCES

[1] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proceedings of the 18th ACM Symposium on Operating System Principles (SOSP)*, 2001.

[2] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman. Planetlab: an overlay testbed for broad-coverage services. *SIGCOMM Comput. Commun. Rev.*, 33(3):3–12, 2003.

[3] S. Guha and P. Francis. Characterization and measurement of tcp traversal through nats and firewalls. In *Internet Measurement Conference 2005 (IMC '05)*, 2005.

[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] A. S. M Steinder. Increasing robustness of fault localization through analysis of lost, spurious, and positive symptoms. In *Proceedings of INFOCOM*, 2002.

[7] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. User-level internet path diagnosis. In *Proceedings of ACM SOSP*, 2003.

[8] 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.

[9] L. Wang, K. Park, R. Pang, V. Pai, and L. Peterson. Reliability and security in the codeen content distribution network. In *Proceedings of the USENIX 2004 Annual Technical Conference*, 2004.

[10] A. Ward, P. Glynn, and K. Richardson. Internet service performance failure detection. *SIGMETRICS Perform. Eval. Rev.*, 26(3):38–43, 1998.

[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.