*Active Probing Approach for Fault Localization in Computer Networks**

Maitreya Natu
Dept. of Computer & Information Science
University of Delaware
Newark, DE, USA, 19711
natu@cis.udel.edu

Adarshpal S. Sethi
Dept. of Computer & Information Science
University of Delaware
Newark, DE, USA, 19711
sethi@cis.udel.edu

***Abstract*—Active probing is an active network monitoring technique that has potential for developing effective solutions for fault localization. In this paper we use active probing to present an approach to develop tools for performing fault localization. We discuss various design issues involved and propose architecture for building such a tool. We describe an algorithm for probe set selection for problem detection and present simulation results to show its effectiveness. We demonstrate through analysis and experiments that active probing has the potential to greatly reduce the probe traffic and the fault diagnosis time.**

Keywords- active probing; fault localization; active monitoring; probe station selection; problem detection; problem determination;

I. INTRODUCTION

Monitoring techniques are used in computer networks to support a wide range of activities involving network design and operation [7, 9, 17, 18]. Network monitoring can be separated into two broad categories: Active and Passive monitoring. Active monitoring involves sending traffic onto a network to sample its behavior. This traffic is sent in the form of probes which can vary from simple probes such as pings to complex test transactions.

Passive monitoring does not produce additional traffic. Rather it listens to traffic that transits through a particular point on a network. At its simplest, counts are made of packets; in more sophisticated implementations, analysis is done by inspecting packet headers. Passive measurements are mainly used to measure metrics pertaining to a certain network element, e.g., at-a-point metrics such as link throughput, and packet size statistics. However from an application point of view, end-to-end quality of service metrics might be of concern and for these the passive approach is inappropriate as the presence of traffic between the two end-points is not guaranteed. Active monitoring is typically used to obtain end-to-end statistics such as latency, loss, and route availability.

*Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

Besides generating traffic between selected points of interest, there are other advantages of active monitoring. It provides flexibility in the design of probe streams with particular properties to match measurement requirements. For example, after localizing a problem to a particular network point by measuring quantities like average delay and loss on a route, finer probes can be sent to identify the bottleneck and available bandwidth[7, 17], or to estimate cross traffic[16]. The main disadvantage of active monitoring is its invasive character. Probes may modify route conditions and perturb the very traffic one is trying to monitor. To minimize these effects, probe streams of low average bandwidth are used [5].

Active monitoring techniques use probing for a variety of network monitoring applications. As discussed in Section II, probing can be used broadly in two ways: active and pre-planned probing. Active probing has the potential to develop effective solutions for network monitoring applications due to its fundamental end-to-end nature and flexibility. One such application is fault localization. Fault localization identifies the fault that can best explain the observed network disorders. Active probing can be used to perform efficient fault localization where probes can be selected in real time and sent to diagnose the root cause of a failure.

In this paper, we present a general approach to using probing technology discussing various ways of developing such tools and their applications. We then propose architecture for using active probing to perform fault localization in networks. Active probing solutions for performing fault localization involve three main steps: probe station selection, problem detection, and problem determination. We discuss various design issues involved in probe station selection. We then discuss various factors affecting probe set selection for problem detection and determination. We develop an algorithm to select probes to perform problem detection and present simulation results to show the accuracy and effectiveness of the proposed algorithm. We demonstrate through analysis and experiments that active proving can be a powerful and effective technique for fault localization.

II. GENERAL APPROACH TO PROBING

Probing is used in network monitoring in a variety of ways and for a variety of purposes [9,14,19,20]. Probing is an

information gathering approach, performed by using test-transactions sent by a probe station to the nodes in the network under observation. The probe results are then analyzed to infer the state of the network. Network parameters and conditions can be inferred from probe results, e.g., the variance in delay, loss percentage etc.

Probing has been used in network monitoring applications broadly in two ways:

- **Preplanned probing:** It involves offline selection of a set of probes [3]. These probes are periodically sent out in the network. This is followed by a passive data-mining approach to infer the network state by analyzing the probe results. This approach generates a large amount of management traffic, a large part of which might be wasteful. Another significant drawback in this approach is the involved difficulty in envisaging all possible problems and generating a probe set for it. Also as the probes are sent at periodic intervals of time, the inference procedure can involve a delay. The involved delay can cause a certain degree of inaccuracy in the network state inferred from the probe results. Preplanned probing, however, imposes less overhead on the manager for selecting probes.

- **Active probing:** It adapts the probing strategy to the observed network state. Instead of sending probes for locating all potential problems in the network, it sends a minimal number of probes initially and then adapts the probe set to the observed network state [3, 19]. The probe stations then send probes that provide most information gain. This approach can greatly reduce management traffic and provide more accurate and timely diagnosis. Key goal of active probing based network measurement is to be able to obtain accurate, reliable estimates using only a small number of probes and using probe streams of low average traffic.

A. Probe Types

Various types of probes have been used in the past for monitoring different network characteristics.

- **1-packet:** V. Jacobson in his 'pathchar' tool used 1-packet method, to estimate link bandwidth from round trip delays of different sized packets from successive routers along the path [6]. One packet methods are based on the assumption that the transmission delay grows linearly with the packet size.

- **Packet pair:** These methods are based on spacing effect of the bottleneck link. They use the minimum inter-departure time of consecutive packets sent back-to-back on a link to estimate the bottleneck bandwidth. Some methods estimate available bandwidth based on the observation of inter-departure time of consecutive probe packets [1,4].

- **Packet train:** A sequence of packet pairs is called packet train. Different methods vary in their use of packet trains based on how the packet pair gaps are controlled by the sender, Methods like pathload [10], IGI, PTR [8] use packet trains with uniform intervals. In contrast in PathChirp [18] and Spruce [23], packet intervals are statistically constructed, forming a non-uniform packet train.

- **Packet tailgating:** This method uses packet trains consisting of large packets interleaved with small tailgating packets. Large packets exit midway due to limited TTL but small packets travel to the destination while capturing important timing information. Many packet dispersion based bandwidth tools have been developed in the past [10]. They are based on self-induced congestion. Probe packets temporarily induce network congestion if and only if the probing bit rate exceeds the available bandwidth on the path, thus increasing queuing delay significantly. The minimum probing bit rate that causes network congestion hence gives an estimate of available bandwidth.
- **Hybrid methods:** These methods exploit both the 1-packet and packet-pair effects, e.g., Packet Quartet [15] uses packet quartet probe class where probes are replaced by probe and pacesetter pair. Different estimation methods are built on this framework based on delay variation and peak detection.

B. Probing at Different Levels of Granularity

End-to-end active probing is mainly used for behavioral monitoring. Simple behavior like connectivity is monitored by basic tools like ping. There are more complex behaviors that can be monitored such as bandwidth, traffic levels, loss and jitter, path MTU and other characterizations using different types of probes.

Probing can be used to detect SLA violation, which might be defined in terms of response time thresholds, packet loss thresholds etc. Probes to test SLAs can compose of a set of requests to the target application. In order to perform a deeper diagnosis, e.g. to detect the exact bottleneck server, more sophisticated probes can be sent. Such probes can be specifically designed application requests that invoke the specific servers that need to be monitored. For even deeper diagnosis, probes can be sent to test specific EJBs, servlets, SQL queries by sending appropriate test transactions, HTTP requests etc.

Probing can also be used at the system or middleware level, where probes can be sent to identity performance bottleneck at disk, processors, memory or incorrect settings of thread pool, heap size or other parameters.

Simple probes like pings or traceroutes are used to detect network layer failures like link or node failures. Different characteristics of probes like loss, delay etc. can be used to infer various aspects of the network state, e.g. available bandwidth, bottlenecks, lossy links, presence of noise etc.

C. *Applications of Probing*

Existing large scale active measurement programs [11, 12, 24] have used probe traffic to measure connectivity, delay and loss statistics. Methods have also been employed to measure bottleneck bandwidth [7, 17] and available bandwidth. Also detailed statistics of delay and loss measurement can be done using active probing. Measurements collected in the Internet focus on topology, workload, performance, and routing.

Probing can be used to identify the composition of application traffic, packet size distribution, packet inter arrival time, performance, path-length etc. Traffic flow matrices can also be computed using probe results to compute a table indicating traffic flowing from a given source to given destination.

Probes can also be used for tracking and visualizing Internet topology: Tools like skitter [9] use traceroute like probes to identify topology details, e.g., specific backbones, traffic exchange points etc.

Routing behavior e.g., effects of outages on surrounding ISPs, effect of topology changes on Internet performance, consequences of new routing policies, etc. can also be detected using probing. Probing can identify potential areas to improve the network's ability to respond to congestion and potential vulnerabilities in the network.

Probing can provide effective fault management solutions for fault diagnosis in a network. Probing solutions are developed for automated monitoring and management of a network at various layers. In the following sections, we discuss how active probing can be used to build efficient solutions for fault localization. We analyze the cost and benefits of active probing and discuss various design issues to build effective techniques for fault diagnosis.

III. FAULT LOCALIZATION

Fault localization is the process of analyzing external symptoms of network disorder to isolate the faults responsible for the symptoms' occurrences. Fault localization is performed at various layers of the protocol stack. Tools are built to diagnose various symptoms ranging from end-to-end connectivity failure to more sophisticated symptoms like SLA violations. A commonly used approach to problem diagnosis is *event correlation* [22], in which every managed device is instrumented to emit an alarm when its status changes. However this approach requires heavy instrumentation to make each device capable to send alarms. Also, the alarms may not reach the manager due to packet loss or inability of the device.

An alternative approach could be to use probing, where the managers can send probes to network nodes to diagnose network health. These probes are test transactions whose outcome depends on certain network components. Thus success or failure of a carefully selected set of such probes can be used to infer the health of the monitored network components. As discussed in Section II, probing solutions can be built using pre-planned or active probing. Pre-planned probing can be expensive and inefficient for the task of fault localization in terms of the number of probes needed. However, active probing shows potential to build effective solutions for fault localization. In this section we discuss various design issues involved in developing active probing solution for fault localization.

Active probing can be used to generate efficient fault localization solutions. As compared to traditional fault localization, active probing based solutions impose lesser fault management traffic and lesser delay in the fault diagnosis process. Moreover, as the network manager can actively select probes inferring the previous probe results, it can narrow down to a finer granularity in localizing the fault, by selecting probes specific to the localized area of the network.

Figure 1 shows the architecture for an active probing system for fault localization. This system consists of 3 main components: probe station selection, problem detection, and problem determination. Probe station selection module selects the best locations to deploy the probe stations using the available dependency model information about the network routes. Based on the selected probe stations, a set of available probes from these probe stations are identified. Problem detection component selects the smallest set of probes from the available probes, which can be used to detect a failure in the managed network. Problem detection module triggers problem determination when a failure is detected. Problem determination module infers the network state from the observed probe results and the probe's dependency information. It then selects additional probes online to obtain more information. It repeats this process of analysis and selection till the fault localization is complete.

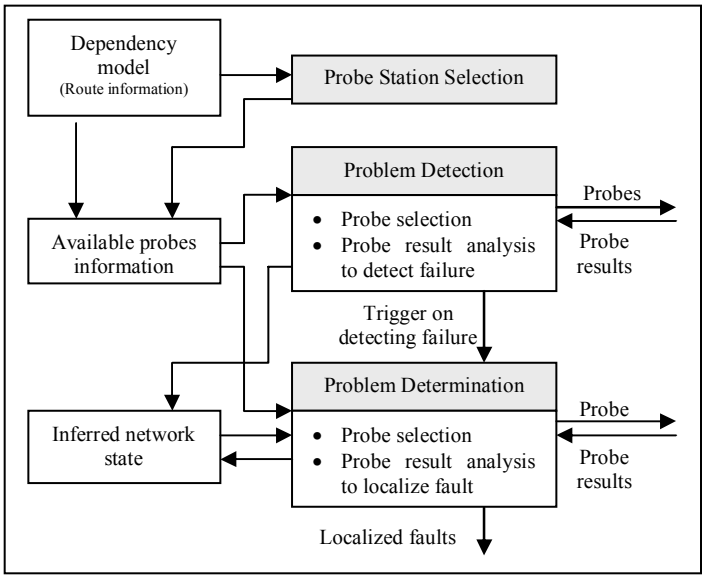

Figure 1: Active probing system architecture for fault localization

Fault localization using active probing involves two steps: problem detection and problem determination. Problem detection is the process of probing the network such that the occurrence of failure of any network component can be detected. Problem determination is triggered when some failure is detected. Problem determination involves analyzing probe results and sending additional probes to determine the exact cause of failure.

- **Problem detection:** During problem detection, probe stations periodically probe the network by sending a pre-selected set of probes. Probe results are analyzed to detect the presence of a fault or performance problem. The pre-selection of probe set for problem detection can be done offline. As these probes are run periodically even when the network is healthy, the probe set should be minimized to impose minimum network management traffic, but still be able to detect all possible problems in the network.

- **Problem determination:** Once some problem is detected by the initial probe set, probe results are analyzed to infer the most probable explanation of the observed probe results. These probe results only provide an indication of some failure in the network, but may not be able to locate the exact cause of failure. Thus observing the probe results, new probes are selected online to obtain more information for performing problem determination. These probes are selected to minimize the time required to diagnose the fault while keeping the extra traffic as low as possible. Moreover the probe selection is done online to select the best set of probes that can give most information for the problem determination process.

Thus developing active probing solutions involves three main steps:

- Selecting probe stations
- Selecting probe set for problem detection
- Selecting probe set for problem determination

IV. SELECTING PROBE STATIONS

Location and responsibilities assigned to probe stations must be decided while building an active probing solution. These decisions are based on nature of routes, nature of targeted failures, availability of dependency information etc. Below we discuss various such factors that contribute to the overall decision making of probe station selection:

- Nature of targeted failures: Probe station selection depends on the nature of faults that need to be diagnosed. Assuming a connected network, to detect a single node failure, a single probe station can be sufficient. However in the same network, to detect an edge failure, we might need more than one probe station because, while the probe paths from a single probe station though can reach all other nodes, they might not cover all the edges. For instance, consider the network shown in Figure 2. Consider node 1 to be a probe station. The bold lines form a spanning tree rooted at node 1 and show routes used by probes transmitted from node 1 to all other nodes in the network. Probe station 1 can detect any single node failure in this network. However, it can detect failure of only those links that are used in reaching the other nodes in the network, i.e., the links shown in bold.
- Maximum number of failures: The assumption of maximum number of faults that need to be detected in a network is an important factor in selecting the probe stations. In a connected network, a single node failure can be detected by just one probe station. However a single probe station might not be sufficient to detect two faults, if both faults occur on the same probe path. For instance, in Figure 2 with node 1 as the only probe station, consider a scenario where nodes 3 and 8 fail. This results in failure of probes from node 1 to nodes 3, 8, and 9. Probe station 1 can only infer failure of node 3 but can not make any inference about the health of nodes 8 and 9.
  Considering the extreme case where all nodes' failure needs to be detected, the probe stations then need to be placed at the vertex cover of the graph formed from the network topology. In that case, all nodes will be one hop away from some probe station, making the closest probe station detect that node's health.

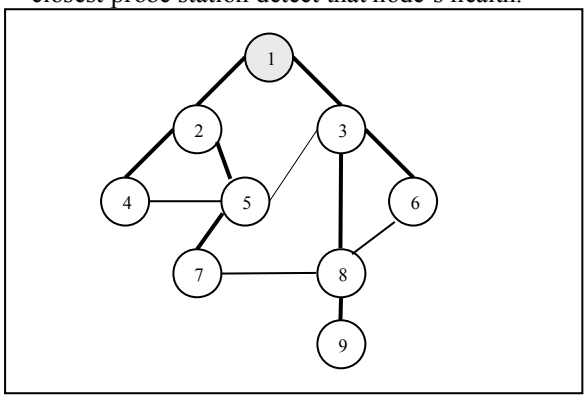

Figure 2: An example network with node 1 as probe station. The bold lines show the links used by node 1 to reach other nodes in the network.

- Probe station failure: The problem becomes even more challenging when the probe station failure is taken into consideration. In case of probe station failure, probe stations must be chosen to provide the ability to detect such failures and make another probe station perform the job of the failed probe station.
- Topological constraints: Another important criterion involved in probe station selection is the topological constraint. The nodes with less connectivity need special treatment. Special topology structures like chains and rings also demand specific probe station placement requirements. One approach to simplify this problem could be to devise a solution by reducing the network into smaller sub-networks connected by such specific network structures like rings, chains, leaves etc.
- Static vs. dynamic probe station instantiations: The probe station selection criteria differ if a probe station location can be selected actively based on current diagnosis requirements. As opposed to static probe station selection, this approach provides more flexibility, but deploying probe stations dynamically on any node might not be possible at all places in the network.
- Nature of routes: Probe station selection is also affected by the nature of routes taken by probes.

Considering source routing, a node can probe another node through multiple routes which enhances its probing capacity. The symmetric or asymmetric nature of routes also provides additional information of the probing capacity of the probes. Special care needs to be taken in the presence of loops. If the routes dynamically change, due to load-balancers, source routing, mobility etc., the probe station might not be able to detect the same faults in the changed routing conditions.

- Dependency information: The amount of routing information available for decision making poses some other practical problems in probe station selection. The accuracy and confidence in the information is expressed through the dependency model. Based on the information available this model could be deterministic or probabilistic. It can be complete or incomplete. Moreover based on the nature of the network, the model might change with time. Changes can occur due to route changes or because of availability of more precise information.

V. PROBE SET SELECTION

Once the probe stations are selected, another important step is the selection of probes. These probes are sent by probe stations and the probe results are analyzed to infer the network health. Note that probes need to be selected such that they should not impose significant network traffic. However for providing uninterrupted services, it is desirable to localize the fault as quickly as possible so that healing measures can be deployed.

The probe selection process relies on the information available about the paths taken by the probes. This information is gathered by route discovery agents and is stored in a dependency model. The dependency model represents the dependency relationships between probe paths and the managed network components that it probes [13, 21]. The nature of this dependency information affects the probe selection decision:

- Deterministic or Probabilistic dependency model: The confidence in the information about the path taken by probes determines the confidence in detecting the possible problem and further localizing the exact failure. Thus if the model is deterministic, the probe set selection process is easier than when the model is probabilistic. With a probabilistic model, in the absence of any deterministic information about the probe paths, the probe set which is most probable to detect the network fault is selected.
- Fixed-Variable dependency model: Various network conditions, e.g. load-balancers, mobility, source routing etc., can cause a change in the routes taken by probes. Moreover new nodes may enter and old nodes may get removed from the managed domain. The probe selection algorithm then needs to be made adaptive to these changes.
- Completeness and accuracy of dependencies: Another important factor that needs attention is the completeness and accuracy of the dependency model. At times route discovery agents might not be able to fetch the compete routes. Moreover in a dynamic environment, the routes may change, bringing an inconsistency between the routes in the dependency model and the actual routes. Different measures can be taken to deal with possibly inconsistent and incomplete routes. For instance, a certain degree of redundancy can be introduced by having multiple probes or probe stations to probe the same node. Another measure could be to associate belief values to the inferred hypothesis and reach conclusion only after significant confidence is obtained. Performing a regular update of the dependency model can also improve the accuracy of the problem determination solution.

A. Selecting Probe Set for Problem Detection

As discussed earlier, the problem of probe selection for problem detection involves selecting a minimal set of probes that can detect health of all managed components. These probes may not be able to pin-point the exact cause of the failure but should be able to detect the occurrence of some failure. The problem of selecting minimum probe set for problem detection is similar to the set cover problem, where components probed by each probe form a set. The goal is to come up with a minimum number of sets that cover all the nodes. The set cover problem is NP-Hard. Moreover it can not be approximated well. The execution time of the optimal algorithm to select minimum probes by analyzing all possible combinations increases exponentially with increase in network size. Thus there is a need to develop efficient solutions to find minimal probe set that is close to optimal and that executes in reasonable time. Such algorithm for probe set selection can be designed using heuristics.

In the past, approximations have been proposed by using certain heuristics [2, 3]. Heuristics can be designed to compute the information gain from selection of a probe. Various measures can be used to compute the information gain, e.g. count of un-probed nodes that are probed by a probe. Information of the topology structure and the routes can also be exploited. E.g., certain nodes in the network are probed by very few probes. Since probe selection to probe such nodes needs to be done from a very limited probe space, it should be done before other nodes that have larger probe search space.

In case of a single probe station, consider a spanning tree with the probe station as the root and where the branches describe the routes taken by probes. One approach to probe selection to cover all nodes in the network could be to send probes from the probe station to the nodes that lie on the leaves of the spanning tree. However with multiple probe stations, this strategy might be wasteful as it might probe same nodes multiple times in the region where the two spanning trees overlap.

Also given the uncertainty in the routing information, a certain number of redundant probes can also be sent that probe nodes that are already being probed by another probe station.

An average, minimum and maximum number of probes probing each network component can be decided by the network manager while doing probe set selection.

B. *Selecting Probe Set for Problem Determination*

The problem determination process is invoked when any of the probes among the probe set for problem detection fails. This failure gives an indication of some fault in the network. However the probe set for problem detection does not have sufficient diagnostic power to localize the fault. Thus during the process of problem determination, additional probes are selected online, based on the analysis of previous probe results. The probe set selection is done actively and with the goal of minimizing the fault localization time. Moreover as the probe selection is done online, it should not take long time for selection. Various challenges arise in this task of active probe selection and the complexity of the process is affected by several factors:

- Maximum number of faults: The assumption of maximum number of faults that can occur in the network affects the problem determination process. E.g. under the simplistic assumption of only a single failure, linearly probing each node on the failed probe path can identify the problem. This process can be optimized by searching for failure in a binary search fashion, i.e. by probing the node on the center of the probe, and based on the probe results, selecting the first half or the second half of the probe path for sending the next probe. Similar strategy can be used to detect two faults by probing the failed probes from both ends.

  For instance, consider the network in Figure 2. On failure of probe from probe station 1 to node 9, nodes on the path can be searched by sending probes to node 3, 8 and 9. The number of probes sent can be reduced by probing in a binary search fashion. This involves first sending a probe to node 8. The success or failure of this probe narrows the search to node 9 or nodes 3 and 8 respectively. Success of probe to node 8 explains good health of node 3 and 8 and infers failure of node 9. On the other hand failure of probe to node 8 explains failure at either node 3 or node 8. Another probe to node 3 can localize the fault to node 3 or node 8.

  However to detect larger number of faults, more complex strategies need to be adopted. Optimized strategies can be devised for this problem, based on the nature of probes and the faults.
- Spurious probe failures: The observation of network state is frequently disturbed by the presence of spurious symptoms which are caused by unreliable communication, intermittent network faults, or by overly restrictive thresholds. Such situations can decrease the accuracy of the problem determination process. The problem determination algorithm tries to find explanation of all these spurious symptoms, thereby creating explanation for many non-existent faults.
- Nature of available dependency information: The analysis of probes to infer the network state depends very much on the nature of dependency information. As discussed before, the dependency model stores information about the relationship between probes and network elements that are probed. The correctness and completeness of this dependency model affects the accuracy of problem determination. Many route discovery agents fail to provide complete routes due to certain network conditions. In a dynamic environment, e.g. in a MANET, due to mobility the routes taken by probes change. These route changes introduce inaccuracy in the previously built dependency model[13]. Thus building and maintaining the dependency model is a challenging task. The problem determination process needs to be robust against such dependency model limitations.
- Old symptoms: A probe result explains the current network state. However with time the network state changes. This change could be because of mobility, healing measures, entry of new nodes in the network etc. Thus after some time a probe result gets old and if still considered in the problem determination process, can adversely affect its accuracy.
- Storage of probe results: The network state inferred from probe results can be stored in different ways. Certain event driven techniques maintain state by encoding partial problem determination results derived from the observed probe results. This provides a condensed network state representation. However such a representation looses the information of each probe result contributing to the overall diagnosis. Instead, maintaining information about the sequence of individual probe results that contribute to overall fault diagnosis can help refine the results with time when more accurate information is available. Such state representations allow deleting old and spurious symptoms and updating the dependency information of previously observed symptoms [13].

C. *Problem Detection Algorithm*

In this section, we present an algorithm for optimizing the selection of probes for detecting the occurrence of a fault in the network.

1) *Challenges involved*

- The problem of finding a minimal set of probes covering all the network components is precisely the Set-Cover problem, which is not only NP-Hard but also can not be approximated well.
- With increasing size of the network, the number of available probes also increases. This increases the search space to select the minimal set of probes.
- In practice, it might not be possible to accurately obtain the route followed by each probe. Thus a probe success or failure might not have a deterministic relationship to the success or failure of the network elements it is expected to probe. This relationship between probes and network

components then needs to be explained using a probabilistic model.

- Probe analysis becomes more challenging in presence of incomplete and inaccurate dependency information. Issues like transient failures, spurious symptoms, dynamic routing, load balancers, node mobility further aggravate the problem.

2) Previous work

The minimal set of probes covering all the nodes can be found by exhaustive search. All feasible combinations of probes can be explored until the minimal set is reached. This algorithm though optimal has a very high computational complexity, making it prohibitive to be deployed practically. This algorithm can be used only for very small networks.

Rish et. al. [2, 3] propose some general approaches to be used for both probe set selection for problem detection and problem determination. These approaches attempt to find minimal set of probes using certain heuristics. [2] proposes subtractive search that starts with all probes, considers each probe in turn, and discards it if it does not add to the diagnostic capability of the probe set. The approach is faster but its effectiveness in finding minimal set depends very much on the order in which probes are explored. Another approach proposed in [2] is additive search, where at each step the probe giving most informative decomposition is added to the probe set. [19] proposes can active probing approach to select probes for problem detection by incrementally selecting probes that cover the nodes that are not yet covered.

In [25], a technique is presented to integrate passive and active fault reasoning in order to reduce fault detection time, improve diagnosis accuracy, and to minimize the intrusiveness of fault reasoning. If the passive reasoning is insufficient to explain the problem, the proposed approach selects optimal probing actions to obtain better explanation.

3) Probe set selection criteria

Probe set selection criteria for problem detection varies from that for problem determination. Probe set for problem detection is selected such that all network elements are probed. However problem determination requires a probe set that uniquely diagnoses every network element. Probes for problem detection are sent periodically and thus the management traffic produced should be low enough that it does not affect the performance of other applications. Moreover the time-constraints on probe set selection for problem detection are less stringent than that for problem determination. Problem determination is done only when some problem in encountered. Thus probes for problem determination should be selected such that fault localization can be done in minimum amount of time.

We propose a Greedy approximation algorithm that explores the information contained in the dependencies between probes and network components. The algorithm selects the network element which is probed by least number of probes, using the dependency information between probes and probed elements. Out of all the probes probing element *n*, the algorithm selects the one which goes through maximum number of nodes that are not yet probed.

Different nodes are probed by different number of probes, depending on the routes used. Nodes that are probed by less number of probes narrow down the search space for probe selection. Consider the case where a node *n* is probed by only one probe. In this case, the only probe probing node *n* must always be selected, irrespective of the number of nodes it covers. Consider another case, where only two probes pass through a node *n*. Then one of the two probes must be selected to cover node *n*. In this situation, the probe covering a larger number of uncovered nodes is the better choice. This leads to the algorithm presented in Table 1.

As an example, consider the matrix in Figure 3, where rows represent probes and columns represent nodes. Cell (*i,j*)=1 indicates that probe i probes node *j*. In this matrix, node 1 is probed by only one probe, i.e., probe C. Thus probe C must be selected. Nodes 1 and 5 are probed by probe C. Out of remaining nodes, i.e., nodes 2, 3, and 4, node 2 is probed by least number of probes (probe A and B). Thus next probe should be selected to probe node 2. Probe A covers 2 non-probed nodes while probe B covers 3 non-probed nodes. Thus probe B is a better choice.

	1	2	3	4	5
Probe A	0	1	0	1	0
Probe B	0	1	1	1	0
Probe C	1	0	0	0	1
Probe D	0	0	1	1	1
Probe E	0	0	1	0	1

Figure 3: Matrix representing dependencies between probes and nodes such that cell(*i,j*)=1 infers that probe *i* probes node *j*.

We assume a deterministic and complete dependency model for this work and aim to relax this assumption in the continuing work. We present results to show the effectives and execution time of the Greedy algorithm. We first compare the algorithm with the Additive search presented in [2] and the Exhaustive search algorithm. Because Exhaustive search can not run on network with large number of nodes, we ran the three searches on a network with 8 nodes. We varied the average node degree from 2 to 5, setting the maximum node degree to 8. We observed the size of probe set computed by each algorithm. It can be seen from Figure 4 that the size of probe set computed by the Greedy algorithm is close to optimal and is smaller than that computed by the Additive algorithm. Figure 5 shows the time taken by the three searches in doing these computations and it can be seen that the Greedy algorithm takes significantly less time than the Exhaustive algorithm. The time taken by Greedy algorithm is comparable to that of the Additive algorithm.

TABLE I. ALGORITHM FOR PROBE SET SELECTION FOR PROBLEM DETECTION

1. Inputs
   a. N: The set of nodes
   b. PS: The set of probe stations
   c. AvailableProbes: Set of probes that can be sent from probe stations to other nodes in the network
2. Initialization:
   a. SelectedProbes = Null
   b. NonProbedNodes = N
   c. For each probe p ∈ AvailableProbes, Nodes(p) = the set of nodes ∈ NonProbesNodes probed by p
   d. For each node j ∈ NonProbedNodes, Probes(j) = the set of probes ∈ AvailableProbes that probe node j
3. For each node k ∈ PS
   a. Remove k from NonProbedNodes
4. Select node l ∈ NonProbedNodes, that has smallest |Probes(l)|
5. Select a probe q ∈ Probes(l), that has largest |Nodes(q)|
6. Remove each node m ∈ Nodes(q) from NonProbedNodes
7. Remove probe q from AvailableProbes
8. Add probe q to SelectedProbes
9. For each probe r ∈ AvaialbleProbes, update Nodes(r) with the modified set of NonProbedNodes
10. For each node n ∈ NonProbedNodes, update Probes(n) with the modified set of AvailableProbes
11. If ((NonProbedNodes ≠ Null) & (AvailableProbes ≠ Null))
    a. Repeat steps 4-11
12. If (NonProbedNodes = Null)
    a. Exit and return the SelectedProbes
13. If (AvailableProbes = Null)
    a. Exit

To test the algorithm on networks of larger sizes, we ran the Greedy and Additive algorithm on networks with 50 nodes, by varying the average node degree from 3 to 6, with maximum node degree of 7. The comparison of the two algorithms is shown in Figures 6 and 7. Figure 7 shows that execution time of the Greedy algorithm is significantly less than the Additive algorithm; while it can be seen from Figure 6 that the probe sets computed by the Greedy algorithm are smaller than those computed by the Additive algorithm.

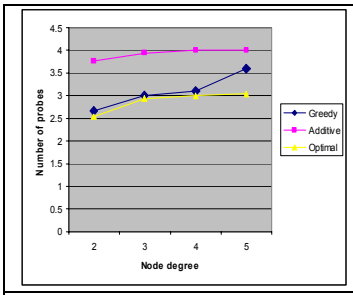

Figure 4: Comparison of probe set size computed by Greedy, Additive, and Optimal algorithms, for a network with 8 nodes varying the average degree from 2 to 5

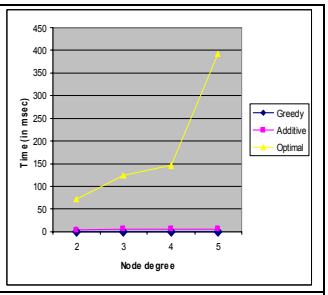

Figure 5: Comparison of execution time of Greedy, Additive, and Optimal algorithms, for a network with 8 nodes varying the average degree from 2 to 5

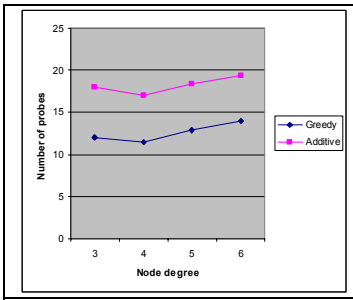

Figure 6: Comparison of probe set size computed by Greedy and Additive algorithms, for networks with 50 nodes.

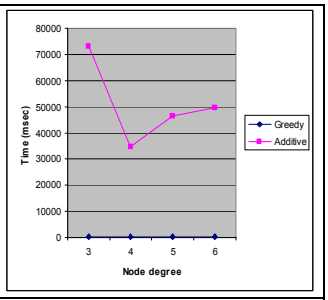

Figure 7: Comparison of execution time of Greedy and Additive algorithms, for networks with 50 nodes.

VI. CONCLUSION

This paper addresses development of active probing solution for fault localization in computer networks. We presented architecture for building such solutions and discussed various design issues for probe stations' selection and probe set selection for problem detection and determination. As our initial work we presented an algorithm for problem detection using active probing. We also presented simulation results to show the accuracy and effectiveness of the proposed algorithm. Analysis and experiments show that active probing can greatly reduce the number of probes and the time required for fault localization as compared to the traditional techniques.

Directions for future work include developing algorithm for probe station selection and real-time diagnosis algorithm for problem determination. We also aim to develop algorithms that can work with probabilistic, incomplete, and inaccurate dependency model. Active probing offers significant potential for problem determination, as well as other network monitoring applications.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied of the Army Research Laboratory or the U.S. Government.

REFERENCES

[1] J.C. Bolot. Characterizing end-to-end packet delay and loss in the internet. *High-Speed Networks*, 2(3), 1993.

[2] M. Brodie, I. Rish, and S. Ma. Optimizing probe selection for fault localization. In *Distributed Systems Operations Management*, pages 1147-1157, 2001.

[3] M. Brodie, I. Rish, S. Ma, G. Grabarnik, and N. Odintsova. Active probing. Technical report, IBM, 2002

[4] R. L. Carter and M. E. Crovella. Measuring bottleneck link speed in packet-switched networks. *Performance Evaluation*, 27&28:297-318, 1996.

[5] L. Cottrell. Comparison of some internet active end-to-end performance measurement projects. In *SLAC*, Jul. 1999.

[6] A.B. Downey. Using pathchar to estimate internet link characteristics. In *ACM SIGCOMM, Cambridge, MA,* 1999.

[7] N. Hu and P. Steenkiste. Towards tunable measurement techniques for available bandwidth. In *Bandwidth Estimation Workshop (BEst03), San Diego, CA,* 2003.

[8] N. Hu and P. Steenkiste. Evaluation and characterization of available bandwidth probing techniques. *IEEE JSAC Special Issue in Internet and WWW Measurement, Mapping, and Modeling,* 21(6), Aug., 2003.

[9] B. Huffaker, D. Plummer, D. Moore, and K. Claffy. Topology discovery by active probing. In *Symposium on Applications and the Internet, Nara, Japan,* Jan. 2002.

[10] M. Jain and C. Dovrolis. End-to-end available bandwidth:Measurement methodology, dynamics, and relation with TCP throughput. In *SIGCOMM 2002, Pittsburgh, PA,* Aug., 2002.

[11] W. Matthews and L. Cottrell. The PingER project: Active internet performance monitoring for the HENP community. *IEEE Communications Magazine special issue on "Network Traffic Measurements and Experiments",* May, 2000.

[12] T. McGregor, H.-W. Braun, and J. Brown. The NLANR network analysis infrastructure. *IEEE Communications Magazine special issue on "Network Traffic Measurements and Experiments,* May, 2000.

[13] M. Natu and A.S. Sethi. Adaptive fault localization in mobile ad hoc battlefield networks. In *MILCOM'05, Atlantic City, NJ,* 2005.

[14] A. Pasztor and D. Veitch. High precision active probing for internet measurement. In *INET '01, Stockholm, Sweden,* 2001.

[15] A. Pasztor and D. Veitch. Active probing using packet quartets. In *ACM SIGCOMM Internet Measurement Workshop,* Nov. 2002.

[16] V. Ribeiro, M. Coates, R. Riedi, and S. Sarvotham. Multifractal cross-traffic estimation. In *ITC Specialist Seminar on IP Traffic Measurement, Modelling and Management 2000, Monterey, CA,* Sep. 18-20 2000.

[17] V. Ribeiro, R. Riedi, and R. Baraniuk. Spatio-temporal available bandwidth estimation with Stab. In *ACM SIGMETRICS,* Jun. 2004.

[18] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, pathChirp: Efficient available bandwidth estimation for network paths. In *Passive and Active Measurement Workshop,* 2003.

[19] I. Rish, M. Brodie, S. Ma, N. Odintsova, A. Beygelzimer, G. Grabarnik, and K. Hernandez. Adaptive diagnosis in distributed systems. *IEEE Transactions on Neural Networks,* 16(5):1088–1109, Sep. 2005

[20] I. Rish, M. Brodie, N. Odintsova, S. Ma, and G. Grabarnik. Real-time problem determination in distributed systems using active probing. In *NOMS-2004, Seoul, Korea,* Apr. 2004.

[21] M. Steinder and A.S. Sethi. Probabilistic fault diagnosis in communication systems through incremental hypothesis updating. *Computer Networks,* 45(4):537-562, Jul. 2004.

[22] M. Steinder and A.S. Sethi. A survey of fault localization techniques in computer networks. *Science of Computer Programming, Special Edition,* 53(2):165-194, Nov., 2004.

[23] J. Strauss, D. Katabi, and F. Kaashoek. A measurement study of available bandwidth estimation tools. In *Internet Measurement Conference (IMC) 2003, Miami, Florida,* Oct. 2003.

[24] H. Uijterwaal and O. Kolkman. Internet delay measurements using test traffic: Design note. Technical Report, RIPE-158, RIPE NCC, Jun. 1997.

[25] Y. Tang, E. S. Al-Shaer, and R. Boutaba. Active Integrated Fault Localization in Communication Networks. In *IEEE/IFIP Integrated Management (IM'2005),* May. 2005.