Probabilistic Fault Diagnosis Using Adaptive Probing*

Maitreya Natu and Adarshpal S. Sethi

Dept. of Computer and Information Science,
University of Delaware, Newark, DE, USA, 19716,
{natu,sethi}@cis.udel.edu

Abstract. Past research on probing-based network monitoring provides solutions based on preplanned probing which is computationally expensive, is less accurate, and involves a large management traffic. Unlike preplanned probing, adaptive probing proposes to select probes in an interactive manner sending more probes to diagnose the observed problem areas and less probes in the healthy areas, thereby significantly reducing the number of probes required. Another limitation of most of the work proposed in the past is that it assumes a deterministic dependency information between the probes and the network components. Such an assumption can not be made when complete and accurate network information might not be available. Hence, there is a need to develop network monitoring algorithms that can localize failures in the network even in the presence of uncertainty in the inferred dependencies between probes and network components. In this paper, we propose a fault diagnosis tool with following novel features: (1) We present an adaptive probing based solution for fault diagnosis which is cost-effective, failure resistant, more accurate, and involves less management traffic as compared to the preplanned probing approach. (2) We address the issues that arise with the presence of a non-deterministic environment and present probing algorithms that consider the involved uncertainties in the collected network information.

1 Introduction

Modern network environments impose several challenges on the fault localization problems which include (1) presence of multiple failures, (2) incomplete and inaccurate information about the network, (3) non-determinism in the system structure and its observed state, (4) the demand for fault diagnosis with minimal management traffic etc.

One promising approach to effective and efficient fault diagnosis is *adaptive probing*. Probing based approaches perform network monitoring by sending

* 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.

A. Clemm, L.Z. Granville, and R. Stadler (Eds.): DSOM 2007, LNCS 4785, pp. 38–49, 2007.
© IFIP International Federation for Information Processing 2007

probes to determine if the components are in good health. Since probes generate additional traffic in the network, it is important to carefully select probes such that the desired diagnostic capability can be achieved with less traffic overhead. Adaptive probing addresses this concern by adapting the probe set to the observed network conditions by sending less probes in the healthy areas of the network and more probes where a failure is detected.

The past work on probing relies on a complete, accurate, and deterministic information about the underlying dependencies between the end-to-end probes and the probed components. A non-deterministic model is needed to address the issues that arise when the causal relationships among the system elements cannot be learned with certainty. For instance, if the dependencies change dynamically, or when the information about these dependencies provided to the management system is not guaranteed to be accurate.

A fault diagnosis solution for modern communication systems should have the following properties: (1) Ability to perform reasoning under uncertainty about the underlying dependencies, (2) Diagnosis of multiple failures, (3) Low management traffic overhead, (4) Small deployment cost, (5) High accuracy and low computational complexity.

With adaptive probing, we attempt to meet the above stated requirements of a fault diagnosis tool. We provide adaptive probing solutions assuming the availability of non-deterministic dependency information. We attempt to find multiple failures. Adaptive probing attempts to minimize overhead of probe traffic. Unlike the traditional way of deploying passive monitors over a large part of the network, probing solutions reduce the instrumentation overhead by requiring instrumentation of a smaller number of nodes as probe stations. Adaptive probing is computationally much less complex that the preplanned approach of probe selection. We show through simulation results that the adaptive probing approach provides a high detection ratio and low false positive ratio as compared to preplanned probing.

This paper is structured as follows. We present the related work in Section 2. We introduce the probabilistic dependency model and the system architecture in Section 3. We then present an adaptive and preplanned probing algorithm in Section 4. We present an experimental evaluation of the proposed algorithms in Section 5 followed by conclusion in Section 6.

2 Related Work

Network probing with low overhead has prompted development of many monitoring approaches. Due to space reasons, we survey only those approaches that directly relate to probe selection.

Probing tools proposed in the past consist of connectivity, latency and bandwidth measurement tools such as [3], [4], [5] etc. Li et. al. in [6] propose to use source routed probes to measure end-to-end performance metrics. Bejarano et. al. [1] propose a probe selection algorithm for monitoring network links based on a greedy heuristic of selecting a probe that covers maximum number of uncovered

network components. In the past, Rish et. al. [10] have proposed adaptive probing approach for fault localization. In our previous work, we have presented algorithms for adaptive probe selection in [8], [9] assuming a deterministic environment.

Most of the work proposed in the past suffer from two main limitations: (1) a preplanned approach is used to build a probe set to localize all possible failures in the network. (2) an assumption of availability of deterministic dependency information is made. An important contribution in this paper is to propose an adaptive probing approach for localizing faults while considering the non-determinism present in the system. We present algorithms for probe selection in a non-deterministic environment where the dependencies between the probes and network components are represented using a probabilistic dependency model.

3 System Architecture

Figure 1 presents the proposed system architecture. The two main components of the architecture are probe station selection and probe selection. The probe station selection module finds suitable locations in the network where probe stations should be deployed. As part of our ongoing research, we are working on the probe station selection problem. The probe selection module refers to the selection of probes such that faults in the network can be detected and localized. Using adaptive probing, we divide the probing task into two sub-tasks which we call *Failure Detection* and *Fault Localization*. Through the *Failure Detection* module, we first send a small number of probes that can only detect the presence of a failure in the network. They might not be able to localize the exact failure. Once a failure is detected in the network, we perform *Fault Localization* by sending additional probes over the selected area of the network in an interactive manner to localize the exact cause of failure.

The *Probe Station Selection* and *Probe Selection* modules of the architecture use the dependencies between probes and the nodes through which these probes pass which are stored in a dependency model. To address the uncertainties involved in the dependency information, we propose to use a probabilistic dependency model. Each node n is associated with a probability of its independent failure *p(n)*. The dependency between a probe-path p and a node n is represented with the probability of the causal implication, $P(p|n)$, which represents the probability that the failure of node n may cause failure of probe p. The causal probabilities can be computed in a variety of ways. For instance, in a scenario of multi-path routing, the probabilities could be based on the policy that the routers or load balancers use to select the next hops. In the presence of mobility, if different dependency models are available for different times, then probabilities could be based on the temporal closeness of the failure time of a path and the built time of the dependency models [7]. We represent the fault-symptom dependencies using a matrix where each column represents a fault and each row represents a symptom. In our case, a fault represents a node failure and a row represents a probe. A *cell(i,j)* in the matrix represents the probability

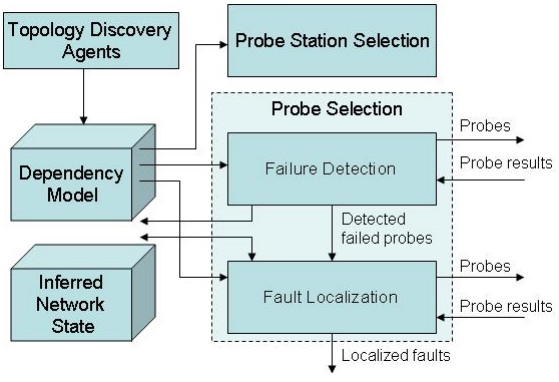

Fig. 1. System architecture for fault diagnosis using adaptive probing

that the node failure represented by column j can cause failure of probe j. In other words, it represents the probability that the probe represented by row j passes through node represented by column j.

4 Probe Selection

In this section, we present algorithms for probe selection using preplanned and adaptive probing. Preplanned probing involves a high computational complexity and we later show through simulation results that preplanned probing is less accurate and requires much larger number of probes as compared to adaptive probing.

4.1 Preplanned Probing

The preplanned probing approach proposes to select a set of probes such that all possible fault scenarios can be uniquely diagnosed. As explained in Section 3, we represent the fault-symptom relationships using a dependency matrix. For the preplanned probing, we extend the dependency matrix such that, along with single-node failures, columns also represent states of combinations of more than one faults that can occur in the system. The algorithm assumes a limit on total number of failures than can be diagnosed, which decides the number of fault combinations that are represented in the matrix. Thus, a $cell(i, j)$ in the matrix represents the probability that failure of fault combinations represented by column j can cause failure of probe i. In other words, it represents the probability that the probe j passes through the nodes represented by column j. For a particular system state corresponding to a column j, the vector of probe outcomes of success or failure would be based on dependency of the probes on the faults represented by the column j. That is, a probe i with high probability value in cell(i, j) is more likely to fail in the given system state as compared to the probe with a smaller probability value. Thus each system state can be represented by a

vector of probe outcomes. The problem of probe selection can then be formulated as finding the smallest set of probes such that each state can be represented by a unique probe vector. In that case, the state of failure can be determined by observing the vector of probe outcomes. In this section, we present a heuristic based algorithm to select such a probe set.

Algorithm PPFL: Preplanned-Probing Fault Localization Algorithm

Initialize partition $statePartition$ = all non-probe-station nodes; set SP = Null;
while *statePartition does not consist of singleton sets* **do**
  Compute $splitOverhead(statePartition, p)$ and
  $splitBelief(statePartition, p)$ for each unused probe p, where
  $splitOverhead(statePartition, p) = \sum_{\forall setS \in statePartition} splitOverhead(S, p)$
  $splitBelief(statePartition, p) =$
  $(\sum_{\forall setS \in statePartition} (splitBelief(S, p)))/|statePartition|$
  where
  $splitOverhead(S, p) = ((|S^-|/|S|)log(|S^-|)) + ((|S^+|/|S|)log(|S^+|))$
  $splitBelief(S, p) = (\prod_{s \in S^+} P(p|s)).(\prod_{s \in S^-} 1 - P(p|s))$
  where S^- and S^+ are the subsets of the set S such that
  $\forall_{n \in S}(P(p|n) > dependencyThreshold) \rightarrow (n \in S^+)$, and
  $\forall_{n \in S}(P(p|n) <= dependencyThreshold) \rightarrow (n \in S^-)$;
  Select a probe p_{min} that minimizes the value
  $0.5.splitOverhead(statePartition, p_{min}) + 0.5.(1 -$
  $splitBelief(statePartition, p_{min}))$;
  Add p_{min} to the set SP;
  $totalSplitBelief(statePartition, SP) = splitBelief(statePartition, p_{min})$;
  while $totalSplitBelief(statePartition, SP) < splitBeliefThreshold$ **do**
    Select a probe q that maximizes the $splitBelief(statePartition, q)$, where
    S^- and S^+ are the sets built by probe p_{min} on splitting each set
    $S \in statePartition$;
    $totalSplitBelief(statePartition, SP) =$
    $totalSplitBelief(statePartition, SP) + splitBelief(S, q) -$
    $(splitBelief(S, q).totalSplitBelief(statePartition, SP))$;
    Add q to SP;
  end
  Divide each set $S \in statePartition$ into subsets S^- and S^+ as computed by
  the probe p_{min};
end

The algorithm starts with partition P consisting of a single set of all possible system failure states under consideration. With each new probe selected, each set in the partition P gets split into two subsets keeping the states that are less likely to fail in one set and those that are more likely to fail in the other set. In the past, work has been done by Brodie et. al. [2] to select probes for fault localization for a deterministic environment (where the dependency matrix only has 0 or 1 values). In this section, we consider a non-deterministic environment and incorporate the probabilistic dependency information in computing the probe set. In what follows, we describe our approach to probe selection for preplanned probing in a non-deterministic environment.

Algorithm GFD: Greedy Failure Detection Algorithm

Initialize the $NodeCoverage(n) = 0$ for each node n; $UncoveredNodes = \{N - ProbeStationNodes\}$;
while $|UncoveredNodes| > 0$ **do**
  foreach *node* $n \in UncoveredNodes$ **do**
    $Entropy(n) = \sum_{(p \in AvailableProbes)\&(P(p|n)>0)} -P(p|n)log(P(p|n))$;
  end
  Select the node *target* with smallest $Entropy(target)$;
  foreach $(probe\ p \in AvailableProbes) and (P(p|target) > 0)$ **do**
    $InformationGain(p) =$
    $0.5 * (P(p|target) - P(p|target) * Coverage(target)) + 0.5 *$
    $\sum_{m \in \{UncoveredNodes - target\}} (P(p|m) - P(p|m) * NodeCoverage(m))$;
  end
  Select the probe p with maximum $InformationGain(p)$; Add p to $FDProbes$; Remove probe p from $AvailableProbes$;
  foreach *node* $n \in UncoveredNodes$ **do**
    $NodeCoverage(n) =$
    $NodeCoverage(n) + P(p|n) - P(p|n) * NodeCoverage(n)$;
    Remove n from $UncoveredNodes$ if
    $NodeCoverage(n) > coverageThreshold$;
  end
end

Algorithm PPFL assumes a *dependencyThreshold* value to consider a probe to be dependent or independent of the failure represented by a certain system state. Thus the dependencies below a *dependencyThreshold* are considered zero and those above the *dependencyThreshold* are considered one. Based on this criteria, a metric can be computed to represent the overhead of the split that can be obtained from a probe. If a probe p splits a set S into subsets S^- and S^+, then the *splitOverhead* for S obtained from p can be computed as:

$$splitOverhead(S,p) = ((|S^-|/|S|)log(|S^-|)) + ((|S^+|/|S|)log(|S^+|)) \quad (1)$$

where $|S^-|/|S|$ and $|S^+|/|S|$ represent the probability that the failure lies in the set S^- and S^+ respectively. $log(|S^-|)$ and $log(|S^+|)$ represent estimates of the number of additional probes required for localization within the subsets S^- and S^+ respectively.

However, since the model is probabilistic, it can not be declared with absolute certainty that failure of nodes represented by some state will not cause failure of probes that have dependencies less than the *dependencyThreshold* value and will surely cause failure of probes with dependency greater than the *dependencyThreshold* value. Hence together with *splitOverhead* we also compute a metric $splitBelief$ to indicate our confidence in the set split obtained by selecting a probe:

$$splitBelief(S,p) = \prod_{s \in S^+} P(p|s). \prod_{s \in S^-} (1 - P(p|s)) \quad (2)$$

The metric represents the belief that a probe does not pass through the nodes represented by states in the set S^- and does pass through the nodes represented by states in the set S^+.

The partition P starts with a single set, and splits into multiple sets in subsequent iterations. Thus the *splitOverhead* and *splitBelief* obtained from a probe p is computed over all the sets of the partition to compute a *splitBelief* and *splitOverhead* value for a partition P obtained by a probe p as follows:

$$splitOverhead(P,p) = \sum_{S_i \in P} splitOverhead(S_i,p) \tag{3}$$

$$splitBelief(P,p) = (\sum_{S_i \in P} splitBelief(S_i,p))/|P| \tag{4}$$

Based on these two metrics, the algorithm selects the probe that minimizes the *splitOverhead* and maximizes the *splitBelief* of the current partition P.

Note that because of the involvement of both factors, the selected probe might not provide an acceptable belief value for the splits of the sets of partition P. That is, failure of p_{min} might not state with enough confidence the success of states in set S^+ and failure of states in set S^-. Hence, we select additional probes to strengthen the belief in the splits. Probes are selected that maximize the *splitBelief* and thus maximize the confidence in obtaining the partition performed by the probe p_{min}. If the probe p_{min} splits a set S into two subsets S^- and S^+, then the *splitBelief* for a set S and thus for the partition P obtained by a probe p in obtaining the same split is also computed using the Equation 2 and 4. Note that the subsets S^+ and S^- derived from the set S are obtained by the split performed by p_{min}. Probe p simply reinforces the belief of getting the same split as defined by p_{min}.

Let SP represent the set of probes selected in the current iteration. We denote as $TSP(P,SP)$ the total belief of getting the split of partition P by the probes in the set SP. Set SP is initialized to p_{min} and $TSP(P,SP)$ is initialized to $splitBelief(P,p_{min})$ When a probe p is added to the set SP, the $TSP(P,SP)$ is updated as:

$$TSP(P,SP) + splitBelief(P,p) - (TSP(P,SP).splitBelief(P,p)) \tag{5}$$

Algorithm PPFL selects additional probes until the $TSP(P,SP)$ value reaches the desired acceptable *splitBeliefThreshold*. At the end of each iteration, each set S in the partition P is split into subsets S^+ and S^- as divided by the probe p_{min}. This new partition is further split in further iterations till all sets in the partition become singleton sets or no probes are left for selection.

4.2 Adaptive Probing

In this section, we present adaptive probing based algorithms for failure detection and fault localization for a non-deterministic environment.

Failure detection. The algorithm is based on identifying the nodes where the uncertainty in selection is minimum, and then applying the Greedy approach of

selecting a probe that gives maximum coverage of nodes among all the probes that pass through this node.

Consider a case where a node n is probed by only one probe. In this case, the only probe probing node n must always be selected in the probe set for failure detection. In a non-deterministic scenario, consider a case where a node $n1$ is probed by 2 probes with probability 0.9 and 0.1 respectively, and another node $n2$ is also probed by 2 probes with probability 0.5 and 0.5. In this scenario, the case of node $n1$ involves less uncertainty making it an easier choice to select the probe that probes node $n1$ with probability 0.9. Hence the algorithm would first choose node $n1$ to cover. In a deterministic environment, a node with minimum probe selection uncertainty can be identified as the node through which least number of probes pass. However, with a probabilistic dependency model, we identify the node with minimum probe selection uncertainty by computing the entropy of the probabilities by which the node is probed by probes [2]. The entropy for a node n is computed as follows:

Algorithm GFL: Greedy Fault Localization Algorithm

foreach *Probe* $p \in PassedProbes$ **do**
  foreach *Node* $n \in SuspectedNodes$ **do**
    if $P(p|n) = 1$, SuspectedNodes -= n; PassedNodes += n;
    if $P(p|n) > 0$, $belief(n) *= \alpha * (1 - P(p|n))$;
  end
end
foreach *Probe* $p \in FailedProbes$ **do**
  foreach *Node* $n \in SuspectedNodes$ **do**
    $PathSuspectedNodes = ProbePathNodes(p) \cap SuspectedNodes$;
    Remove the node $\in PathSuspectedNodes$ from $SuspectedNodes$ and add to $FailedNodes$ if $|PathSuspectedNodes| = 1$;
    if $P(p|n) > 0$, $belief(n) *= \beta * P(p|n)$;
  end
  foreach *Node* $n \in SuspectedNodes$ **do**
    if $belief(n) > failureBeliefThreshold$ **then**
      Remove the node $\in PathSuspectedNodes$ from $SuspectedNodes$;
      Add the node $\in PathSuspectedNodes$ to $FailedNodes$;
    end
  end
end
foreach *node* $s \in SuspectedNodes$ **do**
  Select a probe $p \in AvailableProbes$ that maximizes the $probeWorth(p)$:
  $probeWorth(p) = 0.5P(p|s) + 0.5(\prod_{n \in \{SuspectedNodes - s\}} (1 - P(p|n)))$;
  Remove probe p from $AvailableProbes$; Add probe p to $FLProbes$;
end
Return (FLProbes, PassedNodes, FailedNodes);

$$H(n) = \sum_{(p \in AvailableProbes) \& (P(p|n) > 0)} -P(p|n) log(P(p|n)) \tag{6}$$

where $P(p|n)$ represents the probability that probe p passes through node n. The node with minimum entropy is chosen as the next node to cover.

Once a node n is selected, of all the probes that probe this node, the probe that gives maximum coverage is selected. In a deterministic environment, of all the probes that pass through node n, the probe that passes through maximum number of other nodes can be selected. However, in a non-deterministic environment, two factors decide the probe selection for failure detection: (1) probability gain obtained in covering node n, and (2) probability gain obtained in covering other nodes. For each node m, we maintain a value $Coverage(m)$ to represent the probability that the node m has been covered by the probes selected so far. For probe selection, we compute a metric to identify the improvement that can be obtained in the probability of covering node n and probability of covering other nodes by selecting a certain probe. We represent this metric as:

$$0.5 * P(p|n)(1 - Coverage(n)) + 0.5 * \sum_{m \in \{N-n\}} P(p|m)(1 - Coverage(m)) \quad (7)$$

After selecting the probe p, it is removed from the available probes and the value $Coverage(n)$ is updated for each node as follows:

$$Coverage(n) = Coverage(n) + P(p|n) - P(p|n) * Coverage(n) \quad (8)$$

Any node n with $Coverage(n)$ greater than a $coverageThreshold$ is considered covered and is removed from the node set. The process of probe selection continues till all nodes are covered. Algorithm GFD presents this Greedy algorithm for failure detection.

Fault localization. In this section, we present Algorithm GFL for probe analysis and selection in a non-deterministic environment.

Probe analysis

We use a belief metric to express the confidence associated with a given node failure relative to the failure of other nodes. The belief value is initialized to the probability of independent failure of the node, which we represent by $P(n)$. We update this belief value on observing probe successes and failures. This belief value should not be interpreted as the probability of failure of a node, given the observed probe successes and failures. The belief metric only encodes the relative confidence in the failure of a node in the space of all considered explanations.

On observing the i^{th} probe failure, the belief value of failure of node n, $b(n)$, is expressed using the probability of failure of node n and the probability that node n explains the failure of observed failed probes that have a non-zero dependency on node n. The belief value can be represented as follows:

$$b_{new}(n) = \beta b_{old}(n).P(p|n) \quad (9)$$

where $b_{new}(n)$ and $b_{old}(n)$ represent the new and old belief values for failure of node n respectively.

On observing a successful probe p, we incorporate the probe success information in the belief computation as follows:

$$b_{new}(n) = \beta b_{old}(n).(1 - P(p|n)) \quad (10)$$

The component $(1 - P(p|n))$ provides the probability that failure of node n has not caused failure of probe p. This multiplier decreases the value of the belief metric associated with failure of node n.

Probe selection
After the probe analysis, appropriate probes need to be selected that can give best information for further localization. We build a set of $SuspectedNodes$ that consist of all nodes that have non-zero probability of being on the failed probe paths. In Algorithm GFL, for each suspected node s, a probe is chosen that is (1) most likely to pass through node s, and (2) least likely to pass through any other suspected nodes. Such a probe set can quickly localize the health of its target node, due to high probability of passing through the target node and less number of possible explanations of probe success or failure. For each probe p under consideration, we compute a metric $probeWorth(p)$ considering these factors.

$$probeWorth(p) = 0.5P(p|s) + 0.5(\prod_{n \in \{ShadowNodes - s\}} (1 - P(p|n))) \quad (11)$$

where $P(p|s)$ represents the probability that probe p passes through node s, and the term $\prod_{n \in \{ShadowNodes-s\}} (1 - P(p|n))$ represents the probability that the probe p does not pass through other suspected nodes. The probe p with maximum value for $probeWorth(p)$ is selected to probe suspected node s.

5 Experimental Evaluation

5.1 Simulation Model

We simulated various network topologies with different network sizes and node degrees. Let MD, AD, and N represent the maximum node degree, average node degree, and the total number of nodes in the network respectively. Given these three parameters, we create a network of N nodes, randomly introducing N*AD links such that no node has a degree greater than MD, and also ensuring that the network is connected. We conducted experiments on network sizes ranging from 10 to 50 nodes with an average node degree of 4, and maximum node degree set to 10. Because of the involved computational and memory requirements of the preplanned probing approach, we were not able to run the preplanned probing algorithm on larger network sizes. We use a probabilistic dependency model to represent dependencies between the probes and the nodes used on these probes. We build this dependency model by computing multiple paths between nodes and assigning probabilistic dependency weights to nodes on these paths. The nodes on the longer path are assigned lower weight, while the nodes on the smaller paths are assigned higher weight. We try to find two paths between every end-to-end node by running a shortest path and a second shortest path algorithm. We assume that a probe station can always probe the neighbors that are directly connected to it. We present results to localize node failures in the network.

5.2 Simulation Results

Figure 2(a) shows the number of probes sent by both approaches to localize the failure. The probe counts for the two algorithms are almost the same. Note that,

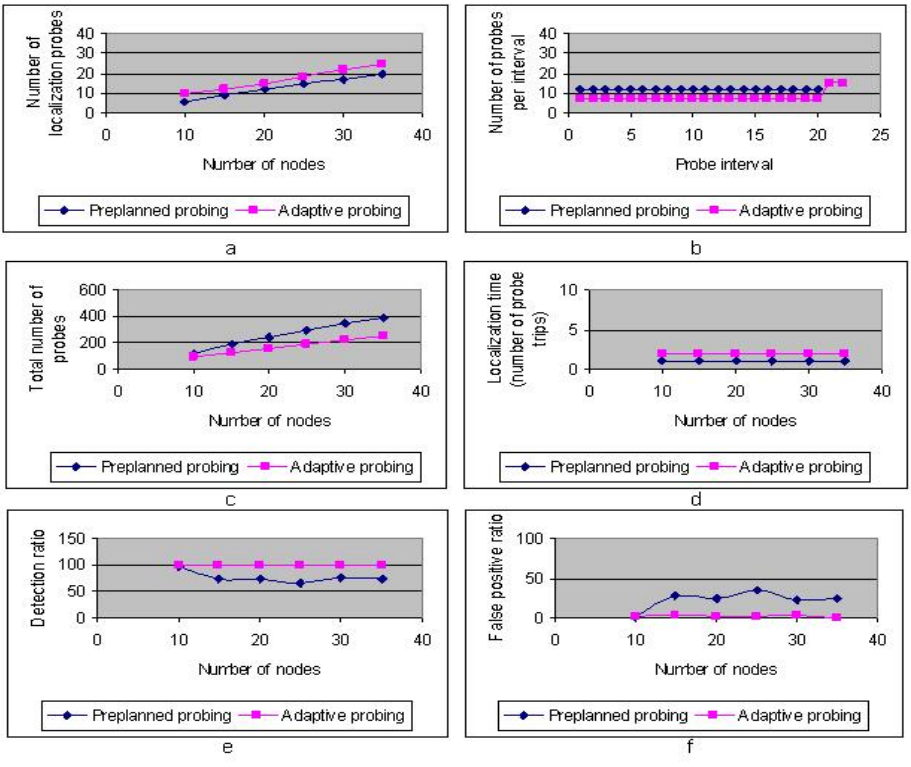

Fig. 2. Comparison of (a) Number of localization probes, (b) Number of probes sent per interval, (c) Total number of probes sent, (d) localization time, (e) detection ratio, and (f) false positive ratio obtained by adaptive and preplanned probing

preplanned probing sends these probes at all times. On the other hand, adaptive probing sends these probes only after detecting a failure. Consider a scenario where a failure occurs in the 20^{th} interval. As shown in Figure 2(b), preplanned probing sends the localization probes in all intervals, where as adaptive probing sends a much smaller set of probes for first 20 intervals, and then sends the larger set of localization probes. As can be seen from the graph in Figure 2(c) which shows the total number of probes sent by both approaches over a period of 20 intervals, the adaptive approach sends significantly less number of probes than the preplanned probing.

Figure 2(d) compares the localization time taken by the two algorithms. Figure shows that adaptive probing is able to localize a failure in almost 2 probe trip intervals which is an acceptable localization time with the amount of decrease in the required number of probes. Moreover, we show through graphs in Figure 2(e) and Figure 2(f), that adaptive probing delivers a higher detection ratio and smaller false positive ratio as compared to the preplanned probing. This improvement can be attributed to the property of adaptive probing that it

can infer the network health from previous probe results to select most suitable new probes, which avoids incorrect diagnosis.

6 Conclusion

In this paper, we presented failure-resistant adaptive probing solutions to monitor a network in a non-deterministic environment. We presented a preplanned probing algorithm to select a set of probes such that all possible failures can be diagnosed. We then proposed adaptive probing based algorithms to select probes for localizing faults in an interactive manner. We showed through simulation results that the adaptive probing approach provides significant improvement in the probe traffic and accuracy of detection over the preplanned probing approach.

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. Bejerano, Y., Rastogi, R.: Robust monitoring of link delays and faults in IP networks. In: IEEE INFOCOM, San Francisco, CA (March 2003)
2. Brodie, M., Rish, I., Ma, S.: Optimizing probe selection for fault localization. In: Distributed Systems Operations Management, pp. 1147–1157 (2001)
3. Downey, A.B.: Using pathchar to estimate Internet link characteristics. In: ACM SIGCOMM, Cambridge, MA (1999)
4. Huffaker, B., Plummer, D., Moore, D., Claffy, K.: Topology discovery by active probing. In: Symposium on Applications and the Internet, Nara, Japan (January 2002)
5. Lai, K., Baker, M.: Measuring bandwidth. In: IEEE INFOCOM 1999, New York City (March 1999)
6. Li, F., Thottan, M.: End-to-end service quality measurement using source-routed probes. In: 25th Annual IEEE Conference on Computer Communications (INFOCOM), Barcelona, Spain (April 2006)
7. Natu, M., Sethi, A.S.: Adaptive fault localization in mobile ad-hoc battlefield networks. In: MILCOM 2005, Atlantic City, NJ (2005)
8. Natu, M., Sethi, A.S.: Active probing approach for fault localization in computer networks. In: E2EMON 2006, Vancouver, Canada (2006)
9. Natu, M., Sethi, A.S.: Efficient probing techniques for fault diagnosis. In: ICIMP 2007. International Conference on Internet Monitoring and Protection, Silicon Valley, CA (to appear, 2007)
10. Rish, I., Brodie, M., Ma, S., Odintsova, N., Beygelzimer, A., Grabarnik, G., Hernandez, K.: Adaptive diagnosis in distributed systems. IEEE Transactions on Neural Networks 6(5), 1088–1109 (2005)