Research Report

Distributed Fault Identification

in Communication Networks

A. T. Bouloutas, S. B. Calo, A. Finkel, I. Katzela

IBM Corporation

T.J. Watson Research Center,

P.O. Box 704, Yorktown Heights, NY 10598

IBM Research Division

Yorktown Heights, New York • San Jose, California • Zurich, Switzerland

Distributed Fault Identification in Communication Networks

A. T. Bouloutas, S. B. Calo, A. Finkel, I. Katzela
IBM Corporation
T.J. Watson Research Center,
P.O. Box 704, Yorktown Heights, NY 10598

June 21, 1994

ABSTRACT:

Telecommunications networks are often managed by a large number of autonomous management centers, each responsible for a particular subarea of the network. In response to a single fault in a telecommunications network, many network elements may raise alarms. These alarms are typically reported only to the management center responsible for the subarea of the network that contains the network element raising the alarm. For a single fault, a particular management center may see only a subset of the alarms actually raised. Management Centers must therefore cooperate in order to identify the underlying fault, and correlate the numerous alarms that may be received. This paper presents distributed fault identification and alarm correlation algorithms which allow these independent management centers to coordinate their activities.

1. Introduction

The dramatic increase in the size and complexity of telecommunications networks in recent years has taxed network operations departments. Techniques for the automated operation and management of telecommunications networks are of increasing interest. One important focus area is fault management. Closely allied to fault management is the problem of alarm correlation and fault identification. In particular, a single fault in a large telecommunications network may result in a large number of alarms, and it is often very difficult to isolate the true cause of the fault. The problem becomes even worse when several faults occur coincidentally in a telecommunications network.

Recently, a number of researchers have proposed algorithms [1, 2, 5, 4, 6] to correlate alarms and identify faults. Some of these techniques are even in the advanced prototyping phase [3].

Unfortunately, all these methods presume that the telecommunications network is managed by a single management center, that has access to all management data. Telecommunications networks are increasingly partitioned into distinct management domains, each managed by an independent management center. This partition may be based on geographical, technological, policy or organizational considerations and the partition may be static or dynamic. The management domains may be disjoint, overlapping, or nested. It is imperative that these fault identification and alarm correlation algorithms take distributed management centers into account.

In this paper we assume that the management of a single telecommunications network is shared by several management centers, each autonomously managing its own domain. Each domain is managed by a single manager process and the domains are disjoint. A domain may be a small, logically autonomous part of a large communications network, such as an Ethernet, FDDI ring, or a Token Ring, or it could be a large subnetwork. The network elements that make up the domain of a manager may or may not in practice be visible to other managers depending on security considerations, interoperability issues, and organizational policy. However, each element of the network is managed by one and only one management center, and the network elements report alarm and historical data to only that management center.

We assume that management centers maintain a peer to peer relationship and there may or may not be a central manager. Manager processes are able to communicate with each other through message passing. A management center need know only how to communicate with management centers that control adjacent management domains. Under this model, a problem that involves network elements from different management domains must be solved collectively by several management centers in a distributed fashion.

Fault identification algorithms are good candidates for distributed processing. In many instances, the effect of a fault is fairly localized, and does not propagate throughout the entire network. In these instances, a distant management center does not need to deal with a minor fault in a far removed network element, unless elements under its control need to communicate with the faulty element, or if the faulty element plays a role in communicating with a another element of interest to it. Therefore, if fault identification processing is distributed, many parts of the network management system will be shielded from information that is not locally useful. Since network management centers tend to overflow with information, this property is very useful.

Typically, faults are detected by network elements and are reported to management centers in the form of alarms and alerts. Once these alarms and alerts are received by a management center, the center has to:

- Correlate the alarms and alerts it has received so far;
- Identify the possible sources of the fault(s); and,
- Oversee the execution of tests to pinpoint the exact location and cause of the fault(s).

The speed of the entire process depends heavily on the first two steps, namely alarm correlation and fault identification. These two steps determine the number of tests to be executed. Testing is generally a difficult and time consuming process. When testing is performed equipment may have to be placed in an off-line state or replacement equipment may need to be installed. Thus, if the network management system is able to correctly identify the source of the fault, the number of tests undertaken will be reduced.

This paper is organized as follows: in Section 2 we present the model and define the problem; in Section 3 we examine the inherent difficulties of distributed fault identification algorithms; in Section 4 we examine a solution for a system which consists of two domains, two regional managers and a central manager; in Section 5 we examine the case of a totally distributed system for two domains; and, Section 6 generalizes the results to many management domains.

2. Model and Problem Definition

In the previous section, we briefly touched on a distributed model for managing telecommunications networks. A telecommunications network is made up of non-overlapping domains. Each domain is managed by a single manager which is solely responsible for that domain. Managers may communicate with other managers on a peer to peer basis via a message passing protocol. However, each manager has a limited view of what is going on in other manager's domains. Thus, each manager has *partial information on the state of the system.* However, each manager has enough knowledge about adjacent managers, and domains, so that communication can be established and messages passed.

Furthermore, we assume that faults may affect network elements but do not affect managers, agents, or management information transfer processes. In practice telecommunications management information systems may have much stricter reliability requirements than other parts of the network. In this setting some problems may affect more than one domain, but these problems do not affect the managers of the domains. The managers of these domains must work together to resolve these problems in a collaborative way.

Network elements often emit alarms or alerts in response to a fault. The network management center in whose domain the element lies has the responsibility of identifying the underlying cause of the alarm or alert and overseeing corrective action when necessary. Each alarm emitted by a network element represents the fault *from that network element's point of view.* Since each network element has only *partial information* about the status of the network, *the alarms emitted by a specific network element provide only partial information about the fault.* The fault identification system must collect all the *partial observations* of the network and *infer* the real state of the network. It is not unusual for an alarm to appear in a network element belonging to a particular management domain and to indicate a fault in a network element in a different domain. Since alarms cross management domains, management centers must collaborate in order to infer the real state of the network.

2.1. Structure of Alarms

Alarms and alerts provide management centers with only partial information about the state of the network. The nature and structure of these alarms is therefore crucial to the fault identification problem.

The *domain of an alarm* is the set of all independent[1] network elements that could have caused the alarm, i.e., the network elements that may be at fault. The domain of an alarm should not be confused with the domain of a management center. A fault may generate a number of alarms. The domain of each of these alarms has to be found in order to correctly identify the fault [1]. The domain of an alarm depends both on the semantics of the alarm and the topology of the communications network. For example if a virtual circuit cannot be

[1] In the probabilistic sense, i.e., a fault in a network element can not cause a fault in some other network element which is independent of the first.

established between two nodes then the domain of the alarm that is emitted in response to this problem includes the two nodes, the virtual circuit, and all the network elements they depend upon. In [1] we propose a normalization of the semantics of these alarms so that the domain of each alarm can easily be determined. General semantics like "upstream," "downstream," "in the network element: *device number*," "in the link: *link number,*" etc., can be used to quickly determine an alarm's domain. In what follows, we assume that alarms are defined only by their domains.

2.2. Information Exchange and Methodology

We assume that each management center is capable of resolving faults that only affect network elements contained in its domain. We will therefore concentrate on the case where faults affect network elements in different management domains.

Once an alarm is emitted by a network element this alarm is reported to the management center. The management center, by interpreting the semantics of the alarm and consulting the configuration database, is able to construct the domain of the alarm. If the domain of the alarm extends beyond the domain of the management center, the information about the alarm has to be sent to other affected management centers as well.

Each management center that receives an alarm is assumed to able to define that alarm's domain within its own management domain.

Assume that at a given time a number of alarms appear in a telecommunications network. We seek algorithms that produce the *"best"* explanation for the alarms that have appeared. Alarms are explained by finding the network element or the set of network elements that could be at fault. Any network element that appears in an alarm's domain is a possible explanation for the alarm. If the domains of two or more alarms have a common intersection, these alarms will have to be examined together because the most likely set of faults that explain all the alarms is not the same as the most likely set of faults that explain each alarm separately. That is why we define the notion of a *cluster* of alarms. A cluster of alarms is a set of alarms whose domains have a non-empty intersection. An alarm belongs to a cluster if its domain has an intersection with the domain of some alarm that belongs to the cluster. Two different clusters do not have a common intersection. We note that a cluster may span more than one management domain since the alarms that comprise the cluster may span more than one management domain.

The alarms in a cluster may have resulted from several faults. The alarms are explained by a set of incidents. An incident represents a set of network elements that may have caused a particular fault and resulted in the emission of a subset of alarms in the cluster. An alarm cluster is explained by one or more incidents.

Often, it is possible to postulate several different incident sets or explanations for a given cluster of alarms. The fault identification algorithm must choose the *best* explanation from the set of possible explanations. One way to find the *best* explanation is to associate with each network element a probability for the two events: the element is in a fault state; or the element is not in a fault state. The best explanation of a cluster of alarms is then the set of network elements whose combined probability of fault is the maximum.

In our model, we consider only independent faults. That is, we assume that the probability of fault in any network element is independent of the state of the other network elements. The combined probability of simultaneous faults at several network elements is then the product of the probabilities of fault of each of the network elements.

More formally we assume that each network element d_j is in one of two states–normal operation or failed state. For each network element we define a binary random variable:

$$d_j = \begin{cases} 1 & \text{if the network element j is operating correctly} \\ 0 & \text{if the network element j is at fault} \end{cases}$$

Furthermore we assume that the d_j are mutually independent random variables with probability distribution:

$$P(d_j = 0) = p_j, \quad P(d_j = 1) = 1 - p_j = q_j$$

If no a-priori information is given about the probabilities of faults then we postulate the *"best"* solution is one that explains all the alarms in a cluster with the smallest number of faulty network elements. This assumption can be justified by assuming that all network elements have the same probability of fault. It is therefore more likely to have few network elements at fault than many.

Instead of associating a probability of failure with each network element (or node) we can associate with it an *"information cost"*, the negative of the logarithm of the probability of failure. Even though information costs are equivalent to probabilities for independent faults, working with information costs has certain advantages. One of these advantages is the intuitively justified additivity property: the probability that two independent network elements (d_1, d_2) are at fault in the network is $P(f) = p_1 \cdot p_2$ while the corresponding information cost is $I(P(f)) = I(p_1) + I(p_2)$. In the rest of the paper we will use the notation $I(\cdot)$ to denote information cost and we will assume that all the network elements (or nodes) have probabilities of fault that are independent of the state of other network elements, thus the information costs (or costs) are assumed to be additive. The focus of the algorithms presented in the rest of the paper is to minimize the sum of the cost of the faulty network elements.

2.3. Measures that should be optimized

As noted in the previous sub-section, problem resolution can be divided into three stages: alarm correlation, fault identification, and testing. The first two stages can be combined into one stage and the problem resolution steps become:

- Fault identification (proposal of various hypotheses); and,

- Testing in order to identify and localize the faults precisely.

Thus, the time to localize a fault is the sum of the time to postulate possible fault hypotheses and the time to test in order to verify the hypotheses. The identification algorithm and the number of network elements in the domain of an alarm cluster affect the speed of the first step. The time of testing is affected by the number of network elements that need to be tested. The workload in a management center is proportional to the number of alarms per unit time, the number of network elements per incident, and the number of tests per unit time. Summarizing, the relevant measures are:

- Number of alarms per unit time;
- Number of network elements per incident; and,
- Number of tests per incident.

A distributed solution improves all the above measures per manager. The number of tests per incident depends on the quality of the initial hypothesis; thus, it depends on how close the cost of the initial hypothesis is to the optimal cost. This will be the basis for the comparison of the distributed algorithms.

3. Why a Distributed Solution is Difficult

Centralized fault identification algorithms are discussed in [1, 2, 5, 4, 6, 3]. We assume the existence of a centralized fault identification algorithm, which can find the most likely faults in a set of network elements given a set of alarms and the information costs associated with the nodes. In [1] we have shown that the fault identification problem is NP-Complete. In general there is no polynomial algorithm that gives an exact solution. One can construct a polynomial algorithm which gives an approximate solution (e.g. see [1]) or a polynomial algorithm that gives the exact solution with some probability. In the Appendix we present one possible probabilistic algorithm which we will use in the rest of the paper. This algorithm finds an exact solution if the number of faults in the network is less than k, a parameter. Thus, it finds an exact solution with probability q, where q is the probability that there are less than k simultaneous faults in an alarm cluster. With probability $1-q$ the algorithm will not output a correct solution. We represent this algorithm by $G(\mathcal{A}, \mathcal{N}, k)$ where $\mathcal{A}$ is the set of alarms, $\mathcal{N}$ is a set of network elements representing the network elements that participate in a cluster of alarms, and k is the maximum number of faulty network elements that can be identified by the algorithm. The development of the rest of the paper is based on the existence of such an algorithm but does not require that the specific algorithm presented in the Appendix be used.

To understand the difficulty of the problem we can use the example of Figure 1. Figure 1 represents 6 network elements, A, B, C, D, E, F and 5 alarms A_1, A_2, A_3, A_4, A_5. Assume that network elements A, B, C belong to a management domain (management domain one) and network elements D, E, F belong to another management domain (management domain two). Alarm A_3 crosses the two management domains. Management domain one can not successfully apply the algorithm $G(\mathcal{A}, \mathcal{N}, k)$ because it only sees three alarms $\{A_1, A_2, A_3\}$. Application of the algorithm to only three alarms would give the optimum solution $\{C\}$ for domain one, which is not included in the correct global solution. The problem here is that alarm A_3, which crosses the boundary of the two domains, should be explained only by domain two. Domain one need only explain alarms A_1 and A_2.

4. A Distributed Solution

As the example in the previous section demonstrates a distributed alarm correlation algorithm must have a scheme which accounts for alarms that cross domain boundaries. Two management centers must be able to reach an agreement as to which management center is responsible for explaining an alarm that crosses a domain boundary. Ideally, the

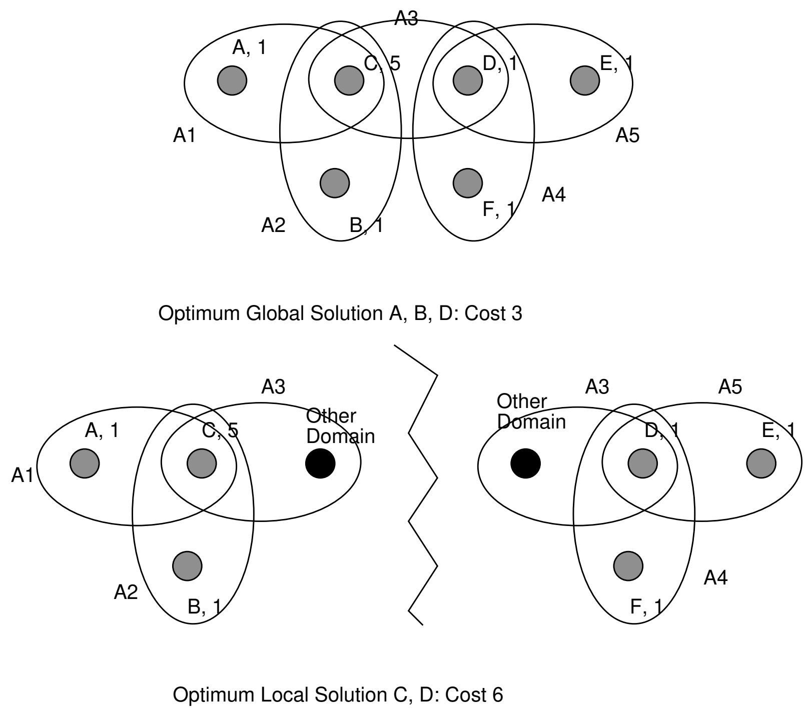

Figure 1: A simple example

management center in whose domain the underlying fault which caused the generation of the alarm lies, should assume responsibility for the alarm. However, assigning an alarm to a management center on this basis is in general NP-Complete. Thus the management centers must reach an agreement based on approximate information. One way to achieve an agreement on this problem is to let each domain *estimate* the cost that would be incurred by explaining each of the alarms. If both management centers make their own estimates for each of the alarms, then they can *"bid"* for the explanation of each of the alarms. Each *"bid"* represents the additional cost that would be incurred if that management center were to explain the alarm.

Returning to the previous example, let us examine the problem from the point of view of domain one. For each alarm that crosses the boundary, domain one would like to know two costs:

- The additional cost that would be incurred if the alarm were to be explained by domain one; and,

- The additional cost that would be incurred if the alarm were to be explained by domain two.

The first cost has to be calculated by management center one, but the second cost has to be communicated by management center two. If domain one knew the bidding cost of domain two for each alarm, then it could represent all the network elements of an alarm that belong to domain two (and thus not seen by domain one) with a *proxy network element.* The proxy network element would represent all the network elements that belong to domain two and could have caused the generation of the alarm. Failure of the proxy network element would indicate that some network element of domain two (among the ones that could have caused the alarm) has failed, thus the alarm is explained by domain two. Note that domain one can associate a cost with the event of a failure of the proxy network element: *The bidding cost of domain two for the particular alarm.* Thus, domain one can solve the centralized problem (using perhaps $G(\mathcal{A}, \mathcal{N}, k)$) and can accurately find the set of faults with minimum information cost in its domain. If the estimation of the cost of the proxy network element is accurate, the solution would be the same as the global one.

Hence, one can apply the centralized algorithm described in the appendix to a new expanded network that includes all the proxy network elements. Here all the alarms that cross the boundary of the domains are treated as regular alarms. Note that in this case if an alarm (among the alarms that cross the boundary between the two domains) is explained by assuming a fault in a network element, this alarm is never explained by a fault in a proxy network element as well. This will be eliminated in the optimization phase. Once the algorithm has output an optimum solution there will be some alarms (among the ones that cross the boundary of the two domains) that will be explained by network elements, and some that are explained by proxy network elements. The alarms that are explained by proxy network elements are the ones that are not explained by domain one and are hopefully explained by domain two. Note that by using this algorithm we impose a penalty on domain one for each alarm that is not explained and is left to be explained by a fault in domain two. Of course a solution which would let domain one explain all the alarms would have larger cost. There is a trade-off and the above described technique can balance the number of alarms explained with the cost incurred to explain them.

4.1. Probability Measures for the Proxy Network Element

It is difficult to calculate the bidding cost to a management center of an alarm that crosses a management domain boundary. The appropriate bid for a specific alarm is one that correctly reflects the added cost to a management center if it is to explain an alarm as originating with a fault at a network element in its domain. If the bidding cost is know a-priori then it is possible to calculate an optimal solution to the fault identification problem. However, exact calculation of an appropriate bid is in general NP-Complete and depends on all the network elements and all the alarms in the cluster. One can only approximate the correct bid. This means that a globally optimum solution can only be found with some probability. Testing must still be relied upon to find the actual cause of a fault. We will not discuss the testing process any further in this paper. Instead, we will examine our solutions to the fault identification problem in terms of how close they are to the optimum centralized solution. This section presents two algorithms for the estimation of the costs of the proxy network elements. Simulation results are used to examine the accuracy of these algorithms.

Algorithm 1

Input:

- A set of alarms that form a cluster C (set of intersecting alarms).
- A set of network elements and the costs associated with each of the network elements.
- An alarm A_1 that crosses two domains, Domain 1 and Domain 2.

Output:

- An estimate by Domain 1 of the bidding cost for alarm A_1.

Method:

- Find the cost of the minimum-cost network element of alarm A_1 among the network elements that belong to domain one.
- This is used as the bidding cost of domain one for alarm A_1 thus is used for the calculation of the cost of the proxy network element by domain two.

With this technique we overestimate the true bidding cost for each alarm. Next we present a more complex algorithm which may give a better approximation of the bidding cost.

Algorithm 2

Input:

- A set of alarms forming a cluster in a domain.
- An alarm, A_1, that crosses domains.

Output:

- The added cost that this domain would incur if it were to explain the alarm, i.e., the bidding cost for alarm A_1.

Method:

- Assign temporary bidding costs according to *Algorithm* 1.
- Have each domain communicate the temporary bidding costs to the other domain, i.e., perform an initial bidding.
- Use the temporary costs assigned to the the proxy network elements by *Algorithm 1* to calculate new bidding costs as follows:
  - Find the cost of the optimum local solution if alarm A_1 is explained by Domain 1.
  - Find the cost of the optimum local solution of alarm A_1 is not explained by Domain 1.
  - The cost for the second case is always less than or equal to the cost of the first. Output the difference of the two costs.

The algorithms presented above give a way to calculate the cost of the proxy network element. The Distributed Fault Identification Algorithm is now presented. It presumes that one of the two algorithms presented above is used to estimate the bidding cost.

Distributed Fault Identification Algorithm

Input:

- A domain accepts as input a set of alarms, some of which cross the boundary of domains.

Output:

- The most likely faults in the domain that may have caused the alarms.

Method:

- *Step 1* For each alarm that crosses the boundary of two management domains, each management center communicates with the other an initial estimate of the probability that this alarm is explained by a fault in its domain. This estimate is based on the probabilities of faults assigned to the network elements in the domain of the alarm.
- *Step 2* For each alarm, given the initial estimates, use $G(\mathcal{A}, \mathcal{N}, k)$ to find the cost differential, i.e. the added cost if this alarm were to be explained by this domain. Communicate this cost to the other domain.
- *Step 3* Accept all the differential costs from the other domains and assign them as costs to the corresponding proxy network elements.
- *Step 4* Apply $G(\mathcal{A}, \mathcal{N}, k)$ to find the optimum solution for the domain (including the proxy network elements.) □

If there are m alarms crossing the boundary between the two domains one can verify that there will be $4m$ messages transmitted from one domain to the other. This is because each domain transmits the initial estimation of the proxy network element and the final estimation. Also one can verify that each domain will apply the algorithm $G(\mathcal{A}, \mathcal{N}, k)$ $2m + 1$ times.

4.2. Simulation Results

A proxy element is placed in the domain of any alarm which crosses a management domain boundary. It is essential to estimate the cost of proxy network elements as precisely as possible. In this Section we compare, via simulation, two ways to estimate the cost of the proxy network elements.

The first approach (*Algorithm 1*) assumes that the cost of the proxy network element is approximated by the cost of the most probable network element to fail among the network elements that are in the portion of the alarm domain contained in a particular management domain. This approximation is fairly crude. For each alarm that crosses the domain boundary of two management domains, Domain 1 communicates to Domain 2 the probability of the network element that is included in the portion of the alarm that is present in Domain 1, and has the highest probability of failure. This probability is used by Domain 2 to find the cost of the proxy network element of Domain 2 for that specific alarm. Similarly Domain 2 calculates the cost of the proxy network element corresponding to Domain 1 and communicates it to Domain 1. This way all the proxy network elements get assigned probabilities (or costs) from the opposite domains.

The second approach assumes that Algorithm 2 is implemented for each proxy network element.

We simulated a system with 70 independent objects. The system was divided into two domains each with 35 objects. Each of these objects could participate in a communication session within its domain or a communication session outside of its domain. The establishment of the sessions took place in a random fashion. The operation of the sessions was assumed to depend on the objects they contain. When an object fails, all the sessions that depend on this object also fail producing an equal number of alarms. Each alarm enumerates all the objects that participate in the session without giving any indication as to which object may have failed and caused the alarm. The simulations assume the existence of network elements that are more probable to fail than others. By default in our simulations 30% of the network elements had high probability of failure and 70% had low probability of failure. The probability of failure of a network element was chosen at random between zero and the maximum probability of failure. For the results presented here, if a network element is among the network elements that are not likely to fail, then the probability of failure was chosen at random between 0 and 0.05. In Figure 3 we compare the two approaches by varying the probability of failure of the network elements with high probability. This varies from 0.05 to 0.20. In Figure 4 we assume that there is an error in the estimation of the actual probabilities of proxy elements when faults occur. We compare the two approaches for different errors in the estimation of these probabilities. In Figure 5 we compare the two approaches for different alarm domain capacities (the maximum number of objects included in an alarm's domain). The alarm domain capacity ranges from 8 to 17. The simulations compare how much the cost of an approach differs from the cost of the optimum solution by calculating the fraction $\frac{Cost(S)-Cost(Opt)}{Cost(Opt)}$. All the results indicate that *Algorithm 2* is significantly better than *Algorithm 1*. Thus, it is important to make an intelligent estimation of the proxy costs. The simulation results should only be considered indications of the correct approach and not a proof since the parameters that can be varied are many and we only examined a small portion of them.

5. Generalization to N Management Centers

The discussion above was predicated on the assumption that the telecommunications network had at most two management domains. Generalization to the case of N management domains involves the same general algorithm. We can distinguish two cases:

- The case where all the alarm clusters participate in at most two management domains (even though there are more than two management domains); and,
- The case where there are clusters that have alarms in more than two management domains.

The first case can easily be mapped into our earlier results. The second case is only a little more complex. The fault identification algorithm that runs in each management center can remain unchanged with the difference that for each bordering management domain a separate proxy agent has to be included. Each of these agents can be assigned a probability indicating the probability that the alarm is explained by a fault in that management domain. The algorithm that runs in in the central management system has to be expanded to combine the results from three or more partial solutions. Given three or more partial solutions the central management system has to verify that the global solution is consistent, i.e., will have to verify that all alarms are explained. The output of the central management system is a set of possible faults that have the minimum information cost and explain the alarms. If each of c management centers produces B possible solutions then the central management system will have to examine at most $O(B^c)$ solutions in order to find the set of partial solutions with the smallest cost that are compatible. Since the number of the possible solutions that need to be examined grows exponentially one could use a pruning method to reduce it. One way to do this is to let the central management center perform the combination algorithm in steps. The first step combines the B best solutions of Domain 1 with the B best solutions of Domain 2. The result, even though it could theoretically include B^2 solutions, can be trimmed down to the best B solutions. Next this result is combined with Domain 3. Again the results can be trimmed down to B solutions and so on.

6. Conclusions

The tremendous growth of telecommunication networks increases the demand for decentralized management. A single network management center may not be able to handle the management requirements of a large network. Large networks are often partitioned into smaller management domains. These management domains must collaborate to solve problems that involve more than one domain. In this paper we have presented algorithms for fault identification for faults that manifest themselves as alarms. The algorithms account for alarms and alerts which cross the boundaries between several management domains. We have proposed two solutions: one assumes the existence of a central management system, while the other assumes a totally distributed environment. For the implementation of the algorithms we have assumed the existence of a centralized probabilistic algorithm for fault identification given a communications network and a set of alarms. This algorithm is described in the Appendix.

A. A Probabilistic Algorithm for Centralized Identification

We present here a probabilistic algorithm for centralized identification. This algorithm can be used in place of $G(\mathcal{A}, \mathcal{N}, k)$. Here we assume that the system will have at most k faults where k is a parameter of the algorithm. In this case the algorithm will give the optimum solution. If there are more than k faults in the system the algorithm will not give a correct solution. Assuming that each network element has a constant probability of failure p then the probability that we have more than k faults in a set of N network elements is given by:

$$Pr(\text{more than } k \text{ faults}) = 1 - \sum_{i=0}^{k} b(i; N, p)$$

where $b(i; N, p)$ is the probability of N Bernoulli trials, with probability p for success, and i successes.

The probability of having more than k faults will give us the probability that the algorithm will not return a solution.

A Probabilistic Algorithm

Input

- A set of network elements N representing the independent network elements that may fail in a communications network.
- For each network element the probability of failure and the information cost $(-\log p.)$
- A cluster of alarms $\mathcal{A}$. Each alarm consists of a set of network elements.
- A number, k, indicating the maximum number of faults that may have caused the alarm cluster.

Output

The set of network elements with the smallest cost that can explain the cluster of alarms.

Method

The algorithm can run in k phases:

- The first phase searches for a single fault that can explain all the alarms and has minimum cost.
- The second phase searches for two faults that could have produced the observed alarms.
- The k^{th} phase searches for a combination of k faults that explain the alarms and have the minimum information cost.
- Output the one solution among the k possible that has the minimum cost. $\square$

In the k^{th} phase there are $\binom{N}{k}$ possible combinations. This number is bounded by N^k, thus this algorithm is bounded by $O(N^k)$. Note that we have assumed that N is much larger than k.

References.

[1] A. Bouloutas, S. B. Calo, and A. Finkel. Alarm correlation and fault identification in communication networks. *IEEE Transactions on Communications. vol 42, no 2/3/4*, February/March/April 1994, pp. 523-533.

[2] Robert H. Deng, Aurel A. Lazar, and Weiguo Wang. A probabilistic approach to fault diagnosis in linear lightwave networks. *Proceedings of the Third International Symposium on Integrated Network Management*, San Francisco, CA, April 18-23 1993.

[3] J.F. Jordaan and M.E. Paterok. Event correlation in heterogeneous networks using the osi management framework. *Third IFIP/IEEE International Symposium on Integrated Network Management*, San Francisco, CA April 18-23 1993.

[4] Marc Riese. Diagnosis of distributed systems: Dealing with incompleteness and uncertainty. *EPFL-DI Rapport technique numero 92/15.*

[5] Marc Riese. Model-based diagnosis of networks: Problem characterisation and survey. *OEGAI-91 Worshop on Model-Based Reasoning, Viennai*, 1991.

[6] Clark Wang and Mischa Schwartz. Fault detection with multiple observers.

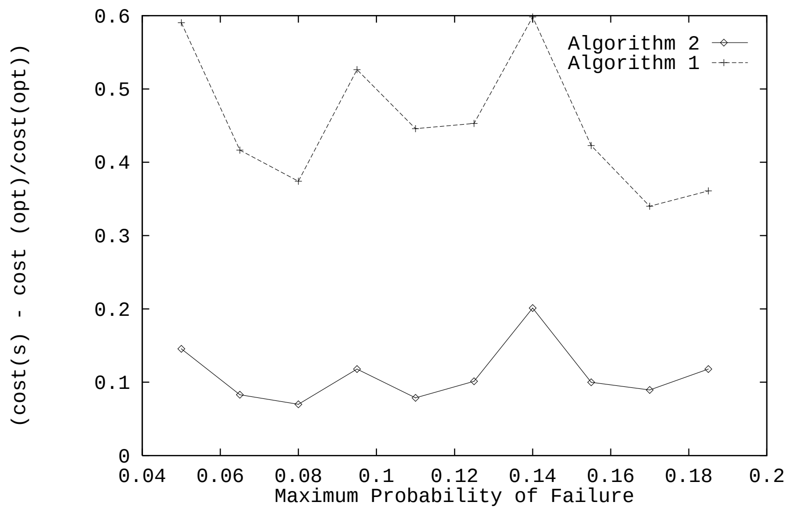

Figure 2: Comparison for different probabilities of failure

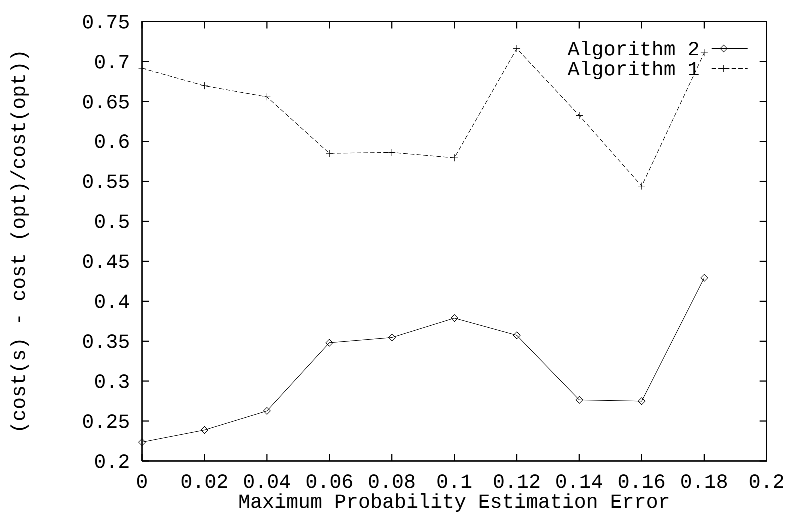

Figure 3: Comparison for different estimation errors

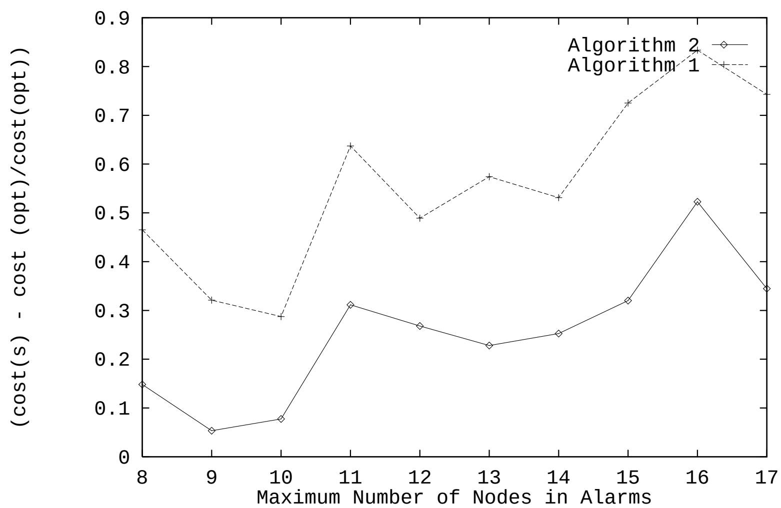

Figure 4: Comparison for different alarm domain capacities