

Locating Faults in Tree-Structured Networks

Christopher Leckie and Michael Dale

Telstra Research Laboratories

770 Blackburn Rd, Clayton

Victoria 3168, AUSTRALIA

Email: {c.leckie, m.dale}@trl.telstra.com.au

Abstract

An important problem in fault diagnosis is how to locate faulty components by analysing performance measurements from those components. In this paper, we present an algorithm that uses the information-theoretic Minimum Message Length (MML) principle to locate faults in tree-structured networks. Tree-structured networks are an important application for fault diagnosis due to their use as distribution networks in telecommunication and power industries. The main advantage of using a MML approach is that we can model both the complexity of an explanation for a set of measurements, as well as the accuracy of the explanation. We demonstrate the effectiveness of this approach through an empirical evaluation on a range of simulated data.

1 Introduction

An important problem in fault diagnosis is how to locate a faulty component in a system by analysing a set of performance measurements from different parts of the system [Doyle 1995, El Fattah and Dechter 1995, de Kleer 1991]. An example of this type of problem is locating faults in tree-structured networks. Networks with a tree topology are widely used in applications such as telecommunications, power distribution and water reticulation, where material is to be distributed from a central site to a large number of consumers. The central site forms the root of the tree, while the consumers are located at the leaves of the tree. In such a network, a fault in one component of the network may affect all consumers connected downstream from that component. By analysing measurements of the quality of service received by each consumer we can infer the location of faults within the network. In this paper, we present an algorithm that uses the information-theoretic Minimum Message Length (MML) principle to locate faults in tree-structured networks.

The complexity of this fault diagnosis problem arises from the need to deal with continuous-valued measurements, and the uncertainty of what measurements are generated by a fault. Fault localisation is trivial when a network component fails completely, since all consumers will report a total loss of service. However, less severe faults can be more common

in practice, and there are several complicating factors that arise from these faults: (1) many types of fault cause a degradation of the quality of service received by the consumers, rather than a total failure; (2) the extent of the service degradation may vary between consumers; (3) even when there is no fault, there can be random fluctuations in the quality of service due to external effects; and (4) at any time, there can be more than one fault affecting the network. Thus, we need an algorithm that can cope with these uncertainties and infer a set of faults that best explains the measurement data.

The main contribution of our work is an algorithm for locating faults in tree-structured networks using the MML principle. The main advantage of using a MML approach is that we can model both the *complexity* of an explanation for a set of measurements, as well as the *accuracy* of the explanation [Georgeff and Wallace 1984]. This means we can determine how many faults are occurring (the complexity of the explanation), in addition to where the faults have occurred (the accuracy of the explanation). In this paper we describe how the MML principle can be applied to this diagnosis problem, and provide an empirical evaluation of the benefits of this approach. We begin in the next section by describing the fault diagnosis problem in more detail. In Section 3, we provide an overview of the MML approach. In Section 4, we describe our fault localisation algorithm and how it uses the MML approach. We then provide the results of an empirical evaluation of our algorithm in Sections 5 and 6. Finally, in Section 7 we compare our approach to related work in the area of fault diagnosis.

2 Monitoring Tree-Structured Networks

In tree-structured networks there is one path from each node to the root of the tree. Figure 1 shows a simple example of such a network containing eight nodes. Consumers are attached to each of the leaf nodes, and the distribution of material is from the root node to the leaves, e.g., from node n_1 to nodes n_3, n_5, n_6, n_7 and n_8 . In general, a fault can occur at any link or node in the network. However, since each node is fed by a single link, a fault in a link is indistinguishable from a fault in a node. Consequently, we can describe the fault diagnosis problem as a search for faults at nodes.

A fault at a node affects the distribution of material through the node, and can thus affect the quality of service

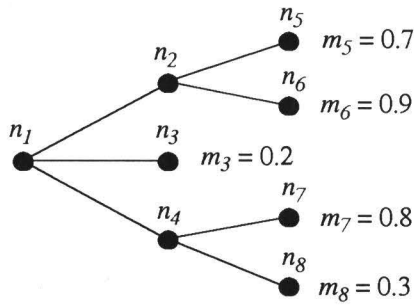


Figure 1: An Example of a Tree-Structured Network

provided to consumers downstream from that node. We say that a node n_d is downstream from a node n_u if there is a path from n_u to n_d and n_u is closer to the root of the tree than n_d . For example, a fault at node n_2 in Figure 1 can affect consumers at nodes n_5 and n_6 but does not affect consumers at any other leaves. We assume that we can measure the quality of service provided to each consumer using monitoring devices at the leaves of the network. These measurements can then be reported back via a separate monitoring network such as the telephone network. We also assume that measurements are taken only at the leaves, and not at any of the internal nodes in the network, e.g., nodes n_1 , n_2 or n_4 . If measurements can be made at the internal nodes then the diagnostic task is considerably simplified. However, this is often not possible in practice, and thus we focus on the problem of locating faults using measurements only at the leaves.

In general, we can assume that an arbitrary number of different types of measurements can be simultaneously made at each leaf. Let m_j represent the j -th measurement at node n_i . We assume that we know the range of values that can be taken by m_j , i.e., the continuous interval $[m_j^L, m_j^U]$, as well as the probability density of these measurements under normal operation $f_j^N(m_j)$, as well as when the node is affected by a fault $f_j^F(m_j)$. In practice, these density functions can be compiled from historical measurements of normal and fault-affected networks.

In Figure 1 we have shown a set of measurements at the leaves of our example tree. For simplicity, we assume that each leaf node in this example has one measurement that is a real-valued number in the range $[0, 1]$, where 0 implies perfect service and 1 implies total disruption of service. These measurements indicate that nodes n_5 , n_6 and n_7 are experiencing a problem.

We need to determine whether these abnormal measurements have been caused by a single fault or multiple faults. For the moment, let us assume that a fault can affect all leaves downstream from the faulty node. One possible explanation for the measurements is that there are three faults in the network $\{F_5, F_6, F_7\}$, where F_i represents a fault at node n_i . An alternative explanation is that there are two faults $\{F_2, F_7\}$, where F_2 causes the abnormal measurements at n_5 and n_6 . Assuming faults are independent, we would assume that the second explanation is more plausible than the first since two faults are more likely than three.

However, the extent to which the measurements at the downstream leaves are affected depends on the corresponding probability distribution. This means that some leaves

affected by a fault may occasionally return a satisfactory performance measurement. Consequently, another possible explanation is that there is a single fault F_1 . This explanation is less likely since nodes n_3 and n_8 are reporting satisfactory measurements, but these measurements can still occur when there is a fault, albeit with a low probability. In terms of the number of faults this explanation is more likely since it involves only one fault, but in terms of the measurements it is less likely because of the measurements at n_3 and n_8 .

The aim of this fault diagnosis problem is to find a set of faulty nodes that best explains the pattern of performance measurements at the leaves. The key issue in this problem is how to decide whether one set of possible faults provides a better explanation of the measurements than another set of faults. This decision function needs to be able to cope with the underlying randomness of the measurements, as well as the probability of having a given number of faults occur. In the following section, we describe the MML approach and its advantages for implementing such a decision function.

3 The Minimum Message Length Approach

The MML approach is a method that uses information theory to evaluate the effectiveness of different theories T_i in explaining a given set of measurements or observations M of the world. It was developed by Wallace as a method of inductive inference [Wallace and Boulton 1968, Georgeff and Wallace 1984], and has similarities to both Rissanen's Minimum Description Length technique [Rissanen 1984] and Bayesian techniques [Cheeseman *et al.* 1988]. In this section, we provide an introduction to the key principles of the MML approach. A detailed tutorial can be found in [Oliver and Hand 1994].

In order to assess how effectively a given theory T_i explains a set of measurements M , we construct a message that describes both the theory, and the measurements in terms of the theory. A theory T_1 is said to be better than another theory T_2 if the total length of the message using T_1 is shorter than the message using T_2 . The justification for this approach is as follows. A good theory should help explain the measurements, and hence the description of the measurements will be shorter than if no theory was used. By adding a description of the theory to the message, we can take into account the complexity of the theory. If the theory is too simplistic, then the theory will have a short description, but the length of the measurements' description will be long. As the complexity of the theory is increased to reflect any structure in the measurements, the length of the measurements' description will shrink. If the theory becomes too complex, the increased length of the theory's description will outweigh any savings in the description of the measurements.

We can encode the complete message as a binary string. By choosing an efficient coding scheme such as Huffman codes, the length of the measurements will reflect the probability of the measurements given the theory, i.e.,

$$\text{length}(M) = -\log_2 P(M / T_i).$$

Similarly, if we can estimate the *a priori* probability of each theory, then the length of the optimum code word for the

theory will be

$$\text{length}(T_i) = -\log_2 P(T_i).$$

Hence, the length of the combined message is

$$\text{length}(T_i + M) = -\log_2 [P(T_i) P(M / T_i)].$$

Note that by Bayes rule, $P(M) P(T_i / M) = P(T_i) P(M / T_i)$. Since $P(M)$ is constant for a given set of measurements, the theory that minimises the combined message length is effectively the theory that has the highest probability given the measurements. In practice, we do not need to construct the actual message since we are interested in the length of the message rather than the message itself.

The MML approach thus provides us with a method for comparing two different explanations of the same set of measurements, assuming that we have an efficient procedure for enumerating the search space of possible theories. Its key advantage is that it allows us to take into account both the effectiveness of the theory in explaining the measurements, as well as the complexity of the theory itself. This addresses the key issues of our diagnosis problem, namely, deciding how many faults are appropriate, and determining the effectiveness of these faults in explaining the measurements.

4 FLoc: An Algorithm for Locating Faults

In this section, we describe our fault localisation algorithm called FLoc. We define in Section 4.1 our structure for the message corresponding to a given set of measurements and possible faults. The length of this message is then used to guide our search for the best combination of faults to explain the measurements (Section 4.2). Let us begin by describing the assumptions FLoc makes about (1) the structure of the trees that it is given, (2) the types of faults that can be located, and (3) the types of measurements generated by these faults.

Trees can have an arbitrary number of nodes N and an arbitrary number of children per node. We have modelled two different types of faults that can affect the nodes of a tree. The first type of fault (F_i^A) corresponds to a major failure at node n_i that affects all nodes downstream. The second type of fault (F_i^L) corresponds to a failure at node n_i that affects only the leaves directly connected to n_i . Such a fault can occur at joints where connections to customers are tapped from the main trunk. For example, if node n_1 in Figure 1 experienced a fault F_1^L , then node n_3 would be the only leaf affected. We denote the default situation where node n_i is functioning normally as F_i^{OK} . The aim of the algorithm is to assign one of these three labels $\{F^A, F^L, F^{OK}\}$ to each node in the network. We assume that any node can experience either fault, although the two faults are equivalent at a leaf node. We also assume that F^A and F^L are equiprobable.

Each leaf has a set of M measurements that are affected by the state of the nodes upstream from the leaf, and we assume that the range $[m_j^L, m_j^U]$ of each type of measurement m_j is known. We also assume that two probability density functions are given for each type of measurement m_j . The first density function $f_j^N(\cdot)$ describes the distribution of possible values for measurement m_j when the leaf node is not affected by a fault. The second density function $f_j^F(\cdot)$ describes the

distribution of values for m_j when the leaf node is affected by a fault either directly or indirectly. We assume that the distribution $f_j^F(\cdot)$ can be used for both types of faults (F^A and F^L).

4.1 Message Structure

To describe completely a given explanation for a set of measurements, we would need to describe the structure of the network, the locations of the faults, and the encoding of the observed measurements given these fault locations. In practice, the structure of the tree is fixed. This means that the tree structure can be ignored since it is a constant overhead in the total message and has no effect in minimising the message length. Consequently, we can divide our message into two parts, namely, the hypothesised faults and the observed measurements.

Faults - When describing a set of hypothesised faults, we can assume that the default status of each node is F^{OK} , i.e., it does not have a fault. This means we need to describe only those nodes where we suspect a fault has occurred. For each node where a fault is suspected, we must encode the identifier for that node and the type of fault that has occurred at the node. If there are N nodes in the tree, then the length of the unique identifier for each node is $\log_2 N$ bits long. Note that since we are interested in the length of the message rather than the message itself, we can allow fractional numbers of bits. If the faulty node is a leaf, then the type of fault is redundant since F^A is equivalent to F^L . Otherwise, we require one additional bit for each faulty node to indicate whether the fault is F^A or F^L .

Measurements - When describing each of the observed measurements, we need to take into consideration whether the measurement is affected by a fault. With this knowledge we can then determine the probability of the measurement by using the appropriate density function, i.e., $f_j^F(\cdot)$ or $f_j^N(\cdot)$. Given that we cannot represent a real-valued measurement with infinite accuracy, we encode each type of measurement m_j to a finite accuracy ϵ_j [Wallace and Boulton 1968]. This is done by dividing the range of each type of measurement $[m_j^L, m_j^U]$ into a set of subintervals of width ϵ_j . We can then approximate a measurement m_{ij} at a node n_i by the value m_{ij}^* , which is the midpoint of the subinterval that contains m_{ij} . We can approximate the probability of m_{ij} as

$$P(m_{ij} / \text{no fault}) \approx \epsilon_j f_j^N(m_{ij}^*) \text{ or } P(m_{ij} / \text{fault}) \approx \epsilon_j f_j^F(m_{ij}^*).$$

Thus, for each node n_i , the length of message describing the j -th measurement is either

- $\log_2[\epsilon_j f_j^F(m_{ij}^*)]$ if there is a fault affecting m_{ij} , or
- $\log_2[\epsilon_j f_j^N(m_{ij}^*)]$ otherwise.

4.2 Algorithm

For a given set of measurements, our algorithm recursively selects the most appropriate fault status for each subtree, starting at the leaves (see Figure 2). At each leaf there are two possible messages: one with a fault and one without. We select the fault status for the leaf based on the shorter of these two messages. We then work our way up the tree by selecting a fault status for each interior node n_i , based on the message lengths of each of the subtrees rooted at the children of n_i . Let us now examine how the message length at an

Algorithm FLoc(n_i)

```

{ if  $n_i$  is a leaf node
  { calculate the following messages:
    message( $F_i^{OK}$ ):
      add measurements  $m_{ij}$  using  $f_j^N(\cdot)$ ;
    message( $F_i^A$ ):
      add fault  $F_i^A$ ;
      add measurements  $m_{ij}$  using  $f_j^F(\cdot)$ ;
    select fault label  $\in \{F_i^{OK}, F_i^A\}$ 
      with the shortest corresponding message;
  }
}
else
{ for each child  $n_c$  of  $n_i$ 
  call FLoc( $n_c$ );
  calculate the following messages:
  message( $F_i^{OK}$ ):
    for each child  $n_c$  of  $n_i$ 
      add faults from the subtree at  $n_c$ ;
      add measurements from the subtree at  $n_c$ ;
  message( $F_i^A$ ):
    add fault  $F_i^A$ ;
    for each child  $n_c$  of  $n_i$ 
      add measurements from subtree at  $n_c$ 
        using  $f_j^F(\cdot)$ ;
  message( $F_i^L$ ):
    add fault  $F_i^L$ ;
    for each child  $n_c$  of  $n_i$ 
      if  $n_c$  is a leaf node
        add measurements from  $n_c$  using  $f_j^F(\cdot)$ ;
      else
        add faults from subtree at  $n_c$ ;
        add measurements from the subtree at  $n_c$ ;
  select fault label  $\in \{F_i^{OK}, F_i^A, F_i^L\}$ 
    with the shortest corresponding message;
  if fault label is  $F_i^A$ 
    set all fault labels in the descendants of  $n_i$  to  $F^{OK}$ ;
}
}

```

Figure 2: Fault Localisation Algorithm

interior node changes based on the fault status that is assigned, i.e., no fault (F_i^{OK}), a fault affecting all children (F_i^A), or a fault affecting only leaf children (F_i^L).

No fault - This is the null hypothesis. If we assume there is no fault at n_i (F_i^{OK}), then the message length at n_i is simply the sum of the message lengths from each child of n_i .

Fault affecting all children - If we assign node n_i the fault status F_i^A , then this fault overrides all faults that were inferred at any of the descendants of n_i . The corresponding message at n_i comprises the new fault label F_i^A and the measurements from all leaves downstream from n_i using the distribution $f_j^F(\cdot)$. Note that if faults had been inferred at most of the children of n_i , then a single fault at n_i is a more concise explanation of the measurements.

Fault affecting only leaf children - If we assign node n_i the fault status F_i^L , then this fault overrides all faults that were inferred at any leaf children of n_i . The corresponding

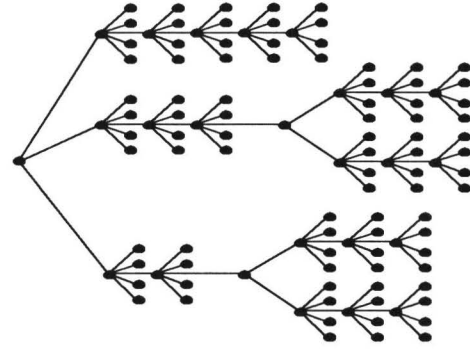


Figure 3: Test Network

message at n_i comprises:

- the new fault label F_i^L ,
- the measurements from all of the leaf children of n_i using the distribution $f_j^F(\cdot)$, and
- the messages from each of the non-leaf children of n_i .

In this way, we provide a more concise explanation for the leaf children of n_i without affecting our explanations for the subtrees belonging to the non-leaf children.

Once we have calculated the lengths of these three messages, we can select the fault status for n_i corresponding to the shortest message. While this fault status is locally optimal for the subtree rooted at n_i , it may be viewed as suboptimal at a higher level node when evidence from the siblings of n_i is taken into account. For example, if we have assigned faults to nodes n_2 and n_3 in Figure 1, we may decide that a more appropriate location for the fault is node n_1 . Assigning a fault F_1^A will override the faults assigned to n_2 and n_3 , and those lower-level faults are then discarded. To simplify the calculation of messages at higher level nodes, we record what the change in message length would be at each node if it is affected by a higher level fault. In this way, the algorithm does not have to backtrack, and its complexity is linear in the number of nodes in the tree.

5 Evaluation

5.1 Aims of Testing

In order to evaluate the performance of our algorithm, we have investigated how the accuracy of FLoc changes under different levels of noise. We used the network shown in Figure 3 as the basis for our tests. It contained 113 nodes, of which 88 were leaf nodes. The structure of this network reflects the types of networks we have encountered in practice. Our two control variables in these tests were the number of measurements per leaf node and the probability density of each measurement.

All leaf nodes have the same number of measurements, and all measurements have the same range and underlying probability density functions. The range of each measurement is the interval $[0, 1]$, with 0 corresponding to normal operation, and 1 corresponding to a total disruption of service. If a node is unaffected by a fault, its measurements have a Gaussian distribution with a peak at 0 and a standard deviation σ ,

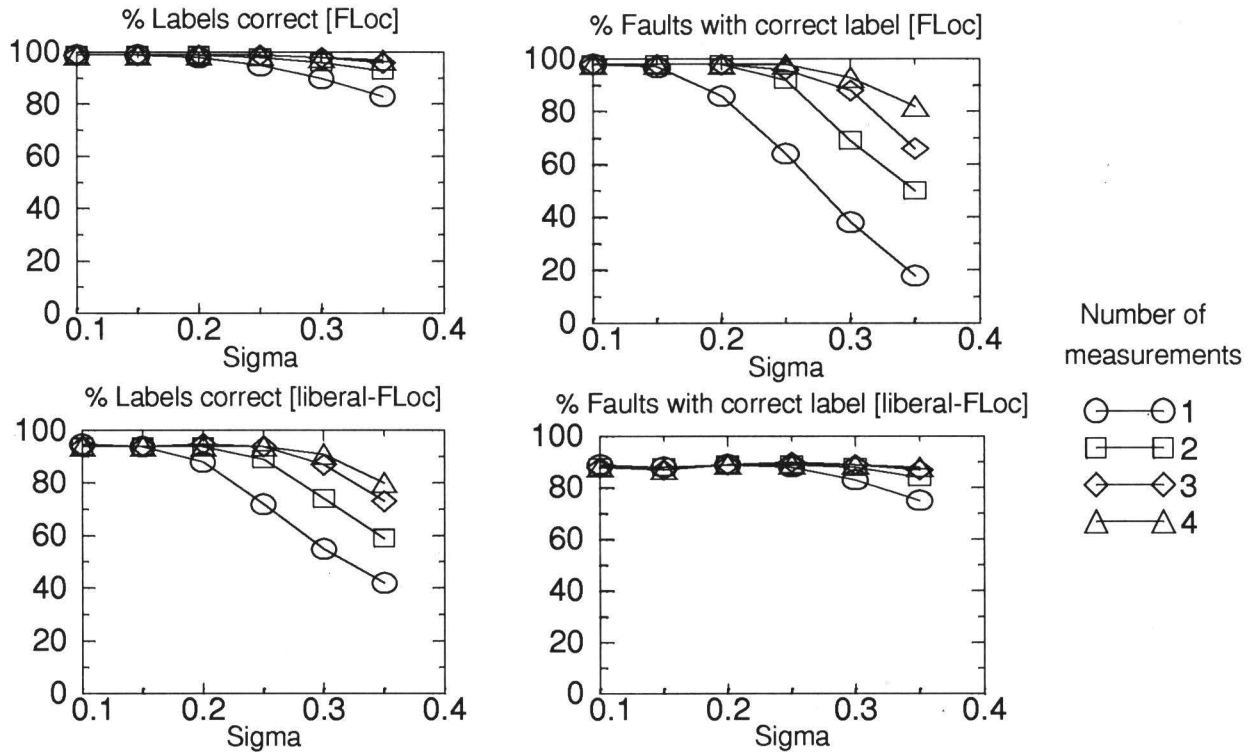


Figure 4: Test Results

$$f^N(m) = \frac{1}{\sigma\sqrt{2\pi}} e^{-m^2/(2\sigma^2)} \quad 0 \leq m \leq 1.$$

The density function for fault-affected measurements is a Gaussian distribution with the peak at 1,

$$f^F(m) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(1-m)^2/(2\sigma^2)} \quad 0 \leq m \leq 1.$$

In our messages, we assume that measurements are encoded with an accuracy $\varepsilon = 0.1$, although the results were generally insensitive to the exact choice of ε .

These measurements were generated by randomly simulating faults in the test network. The probability of any node experiencing a fault of type F^A was set at 0.05, and similarly for faults of type F^L . This allowed us to test the system with multiple simultaneous faults in the network.

By varying σ , we were able to test the sensitivity of our algorithm to noise. A small value of σ means that normal measurements have little similarity to fault-affected measurements. In contrast, a large value of σ means that there is a much greater overlap between the two distributions, thus making it easier to mistake a normal measurement as faulty, and vice versa. We tested FLoc under 24 different parameter settings, allowing σ to vary from 0.1 to 0.35 and the number of measurements per node to vary from 1 to 4. We performed 10,000 trials for each combination of parameter settings, which was found to keep the uncertainty in the results to less than 1%.

To provide a basis for comparison, we also tested a less conservative version of FLoc called liberal-FLoc. This version of the algorithm did not include fault labels for leaves in the message it constructs. Instead, it selects the

fault status of a leaf solely on the length of the encoding of its measurements. This makes liberal-FLoc more likely to assign faults to leaves than the null hypothesis of no fault.

5.2 Results

Figure 4 shows the results of these tests for the FLoc and liberal-FLoc respectively. In these graphs, we use the term "label" to describe an inferred fault at a node. In each case, we show (1) the percentage of inferred labels where the location and the type of label correctly matched the actual fault (F^A or F^L), and (2) the percentage of faults with a correct label in the correct position. The first graph indicates how accurate are the labels reported by each algorithm, while the second graph indicates what proportion of faults are correctly detected. On average, each batch of 10,000 trials required 30 minutes of CPU time on a Sun Sparcserver 20 using an implementation of the algorithm written in Java.

6 Discussion

Let us begin by analysing the performance of FLoc. The vast majority of labels generated by FLoc were consistently correct. When σ was small, nearly all faults were correctly detected. As σ increased, the accuracy of the labels was maintained, but fewer faults were detected. This was due to the increase in the number of faults with borderline measurements, especially at leaf faults. The proportion of faults detected decreased with the number of available measurements, since with fewer measurements, it is more likely that all of a fault's measurements have uncharacteristic values. Thus with noisy inputs, FLoc had a low number of errors of

commission but a high number of errors of omission.

In comparison, when σ was small liberal-FLoc performed 4% worse in terms of the accuracy of its labels, and 9% worse in terms of the number of faults detected. This was mostly due to the tendency of liberal-FLoc to blame leaf faults on the parent of the leaf. This happens because liberal-FLoc is more likely to consider leaf nodes as faulty, and then combine imagined leaf faults with real leaf faults. As σ increased, liberal-FLoc manages to detect most faults, but at the price of generating a large number of labels where there is no fault. This was to be expected, since liberal-FLoc does not assign labels to faults in the messages at leaf nodes. Consequently, it is more willing to accept borderline measurements as faulty. Thus with noisy inputs, liberal-FLoc has the opposite behaviour to FLoc, i.e., a high number of errors of commission and a low number of errors of omission.

Each type of error has a different penalty. Errors of omission mean that consumers suffering from degraded service go undetected. However, these omissions are usually leaf faults that affect only a single consumer. In contrast, errors of commission mean that we are dispatching maintenance staff unnecessarily, which can be a significant expense. Hence, FLoc is clearly preferable when the cost of dispatching staff is high. One possibility is to use both algorithms and give priority to the output from FLoc, then as maintenance staff are available we can investigate the additional faults from liberal-FLoc.

7 Related Work

Senjen et al. [1993] have developed a hybrid approach to locating faults in tree-structured networks using a mixture of rule-based and connectionist techniques. They use preset thresholds to control the propagation of fault probabilities through the network, as well as rules that encode the relevance of each type of measurement to different faults. The main difficulty in their approach is how to select appropriate threshold levels to control the propagation of fault probabilities. By using an MML approach, we have been able to replace these heuristic thresholds with a more systematic algorithm that uses measurable distributions for the probability of each type of measurement.

El Fattah and Dechter [1995] have also addressed the problem of diagnosis for tree-structured networks. Their approach uses constraint satisfaction and optimisation techniques. However, their approach is designed for problems with a symbolic representation, e.g., boolean logic circuits. In contrast, our approach is designed for sub-symbolic domains that involve continuous and probabilistic representations. de Kleer [1991] has developed a probabilistic approach to focus the attention of a diagnostic system for boolean logic circuits. However, it not designed for domains involving continuous measurements.

Doyle [1995] uses a different approach to diagnosing continuous and probabilistic domains based on historical measurements and causal dependencies. He has developed a causal distance measure to detect and relate abnormal frequency distributions of measurements. This technique is specifically designed for time-series measurements, which we

have not addressed. However, Doyle does not address the problem of multiple simultaneous faults. Thus, the two approaches are complementary, and the ability to analyse time-series measurements is an important future direction that we intend to incorporate into our work.

8 Conclusion and Further Work

We have developed an algorithm for locating faults in tree-structured networks, and demonstrated its effectiveness under a variety of test conditions. By using a MML approach for diagnosis, we are able to find the combination of faults that provide the best explanation for a set of continuous and probabilistic measurements. At present, our diagnosis is based on a "snapshot" of measurements at a single point in time. Our next step is to generalise this approach so that we can incorporate historical information in the form of time-series measurements into our model.

Acknowledgements

We hereby acknowledge the permission of the Director, Telstra Research Laboratories, to publish this work.

References

- [Cheeseman et al., 1988] P. Cheeseman, M. Self, J. Kelly, J. Stutz, W. Taylor and D. Freeman. Bayesian Classification. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, pp. 607-611, 1988.
- [Doyle, 1995] R.J. Doyle. Determining the Loci of Anomalies Using Minimal Causal Models. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pp. 1821-1827, 1995.
- [El Fattah and Dechter, 1995] Y. El Fattah and R. Dechter. Diagnosing Tree-Decomposable Circuits. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pp. 1742-1748, 1995.
- [Georgeff and Wallace, 1984] M.P. Georgeff and C.S. Wallace. A General Criterion for Inductive Inference. In *Proceedings of the 6th European Conference on Artificial Intelligence (ECAI-84)*, 1984, pp. 473-482, 1984.
- [de Kleer, 1991] J. de Kleer. Focusing on Probable Diagnoses. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, pp. 842-848, 1991.
- [Oliver and Hand, 1994] J. Oliver and D. Hand. *Introduction to Minimum Encoding Inference*. Technical Report 205, Department of Computer Science, Monash University, Clayton 3168, Australia, 1994.
- [Rissanen, 1984] J. Rissanen. Universal coding, information, prediction and estimation. In *IEEE Transactions on Information Theory*, IT-30(4), pp. 629-636, 1984.
- [Senjen et al., 1993] R. Senjen, M. de Beler, C. Leckie and C. Rowles. Hybrid Expert Systems for Monitoring and Fault Diagnosis. In *Proceedings of the Ninth IEEE Conference on AI for Applications*, pp. 235-241, 1993.
- [Wallace and Boulton, 1968] C.S. Wallace and D.M. Boulton (1968). An Information Measure for Classification. In *Computer Journal* 11(2), pp. 185-194, 1968.