Optimal Diagnosis for Causal Chains

Inderpal S. Bhandari* **Herbert A. Simon****

Daniel P. Siewiorek*

10 September 1987 CMU-CS-87-151

* Department of Electrical & Computer Engineering,
** Departments of Computer Science & Psychology,
Carnegie-Mellon University,
Pittsburgh, PA 15213.
Arpa net [Bhandari] : isb@faraday.ece.cmu.edu
Phone [Bhandari] : 412-268-6638.

Abstract

An optimal algorithm to do probe selection in causal chains is presented. Probes may have different costs of measurement. The algorithm runs in polynomial time.

This research was supported in part by the Ballistic Missile Defense Advanced Technology Center under contract #DASG60-86-C-0015, the Office of Naval Research, under Agreement #N00014-85-K-0008 and the Defense Advanced Research Projects Agency under contract #F-33615-84-K-1520. The views and conclusions reported in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Ballistic Missile Defense Advanced Technology Center, Office of Naval Research, Defense Advanced Research Projects Agency or Carnegie-Mellon University.

Table of Contents

1. Introduction

Probe Selection (PS) is an important facet of any diagnostic program. The problem solved here is to find an optimal algorithm for PS in a causal chain. The word 'optimal' has been used in the literature on diagnosis to designate both locally optimal and globally optimal algorithms. Locally optimal algorithms use some best-first technique to choose *next* the probe that optimizes some metric and they are not generally optimal, although they do provide good heuristics. Globally optimal algorithms choose that *sequence* of probes that optimizes some metric and they are truly optimal. In this work, optimal will be used to refer to the latter. High-level descriptions of devices make use of hierarchies and aggregated variables to 'hide' the overall complexity of the device and simplify the problem solving process. There are no standards for high-level descriptions of systems, but a causal dependency structure has been commonly used to represent systems in qualitative modeling, economics, etc. Examples may be found in [3], [6]. Causal chains are simple structures that are found in such high-level device descriptions.

The advantage of using high-level descriptions of systems is that it provides a means of using the actual structure of the device to do PS. Previous work on optimal algorithms has made little or no use of the structure of the system that is malfunctioning. Typically, it is assumed that the device is an aggregate of components, but the dependencies between components are not modelled. Some structural information is sometimes introduced in terms of the modular structure of the system, but the dependencies between modules at any one level are ignored.

This objection does not apply to heuristic algorithms. However, the fact that there are heuristics that exploit structure and no optimal algorithms to evaluate them against even for simple structures only increases the need for such algorithms to be developed. In fact this point applies also to the so-called information-theoretic heuristics that do not assume any structure. Real systems have complicated structures. It is probably fair to assume that a heuristic that does not do well on simple structures will do no better on real systems. Thus, the development of optimal algorithms for simple structures will be useful in the evaluation of heuristics.

Algorithms for optimal solutions for PS commonly make the assumption that the costs of all probes are the same. This will be referred to as the equal-cost assumption. A simple example is the 'half-split' technique and its generalizations. This gives the "almost optimal" solution for a causal chain under the equal-cost assumption [5]. The "unequal cost" case was identified as an open problem as early as 1960 [1]. It turns out that there exists a polynomial time algorithm to find the optimal solution for the unequal cost case for causal chains. This algorithm is presented in section 3. For more general structures the problem remains open.

Most of the issues raised here can be found discussed in detail in [2] along with useful references.

Examples of optimal algorithms that may be adapted to do PS, but are largely free of structural constraints, may be found in [7]. Another common model for diagnosis that will not be treated here is found in t-fault diagnosability studies [4]. However, some of the ideas developed in this paper may apply to this domain. This lead is currently being investigated.

The rest of this paper is organized as follows. Section 2 defines the model to be used for analysis and the problem statement. Section 3 presents the algorithm along with a proof of its correctness. Section 4 is a discussion on how the algorithm may be improved along various dimensions and of its limitations.

2. The Model

The physical situation that is being modelled is a troubleshooting session in a repair shop that has to service large numbers of the same device/s. Therefore, the parameters of interest are the costs of tests, the relative frequency of failures of different components, and the shortest average troubleshooting time per session. Before the problem can be formally stated, notation must be developed. A causal chain is represented diagramatically in Figure 1.

```
Functions:     f1         f2             fn
             #------>#------># ... #------>#
Nodes:       0        1       2   n-1      n
```

Figure 1. A Causal Chain.

The nodes are possible testing points. Bi-valued measurements can be made at a node, which is either OK or Not-OK. Associated with every node i is a cost of measuring the node, T_i. The costs of measurement may be different for different nodes. A function can be mapped back to a component in the system. Thus if the node i is OK but i+1 is Not-OK, f_i is **faulty** and the component that provides function f_i should be replaced. Also, the node i is Not-OK if any f_j, $j <= i$ is faulty. Associated with every function f_i is a relative frequency of failure p_{f_i}. These represent the fraction of times f_i has been faulty over the total number of faulty functions for some large number of troubleshooting sessions. Therefore the sum of all p_{f_i}'s is equal to 1, and the individual frequencies can be taken to represent the probability that f_i is faulty.

Problem Statement : Given that n is Not-OK and that 0 is OK, find the PS strategy that results in the shortest expected time per troubleshooting session, assuming that there is only a single faulty function. In Section 4, the multiple-fault case is explored.

2.1. Segments of Causal Chains

A **segment** of a causal chain is any continuous portion of the causal chain. Segments will be denoted by [l,r], where l and r are integers designating nodes and $0 <= l < r <= n$. For example, in Figure 1, [0,3],[0,i] and [2,n] are all segments. Associated with any segment is the probability that the fault will lie in that segment. This is denoted by $p_{[l,r]}$ and is equal to p_r - p_l, where p_x is defined as $\sum_{j=1}^{x} p_{f_j}$. p_0 is defined to be 0. Any strategy for fault location proceeds by partitioning [0,n] into successively smaller and smaller segments. Since one test point is chosen at a time, a segment is partitioned into exactly two smaller segments. Thus every strategy can be represented by a **strategy tree**, a binary tree whose elements are segments. Also associated with a segment [l,r], is the time a troubleshooter expects to spend in that segment per troubleshooting session, assuming that s/he is using an optimal strategy to locate the fault in [l,r] and that the strategy tree used has [l,r] as one of its elements. This time will be denoted by $T_{[l,r]}$. A restatement of the problem is to find the procedure that returns $T_{[0,n]}$ as its expected troubleshooting time per session. TopProbe([l,r]) will denote the first element to be probed in [l,r] as part of an optimal strategy to locate a fault in [l,r], $l < k < r$ for some k.

3. An Optimal Solution

3.1. Some Propositions

. **Proposition 0 :** If [l,r] is an element of a strategy tree, the probability that the troubleshooter will localize the fault to [l,r] is $p_{[l,r]}$.

Proof: The proof is straightforward and is omitted.

Proposition 1 : $T_{[i,i+1]} = 0$.

Proof: Trivially true. No troubleshooting remains to be done, as the faulty function has been identified.

Proposition 2 : Given that TopProbe([l,r]) = i, $l < i < r$; $T_{[l,r]} = p_{[l,r]} \times T_i + T_{[l,i]} + T_{[i,r]}$.

Proof: By definition, $T_{[l,r]}$ = expected time spent in [l,r] using an optimal strategy = (expected time to measure node i) + (expected time spent in [l,i] using an optimal strategy) + (expected time spent in [i,r] using an optimal strategy).

The second and third terms are defined to be $T_{[l,i]}$ and $T_{[i,r]}$ respectively. The first term is equal to (probability that node i is measured)$\times T_i$. Node i will only be measured if the fault is localized to the segment [l,r] in the strategy tree. The probability that this occurs is equal to $p_{[l,r]}$, by proposition 0.

Proposition 3 : $T_{[j,k+1]}$ = Minimum [($p_{[j,k+1]} \times T_{j+1} + T_{[j,j+1]} + T_{[j+1,k+1]}$), ($p_{[j,k+1]} \times T_{j+2} + T_{[j,j+2]} + T_{[j+2,k+1]}$), ... ,($p_{[j,k+1]} \times T_k + T_{[j,k]} + T_{[k,k+1]}$)], for $j < k$. The minimum term in this expression also makes it possible to infer TopProbe([j,k+1]).

Proof: The first node to be probed in [j,k+1] has to be a node i, where $j < i < k+1$. By proposition 2 it follows that $T_{[j,k+1]} = p_{[j,k+1]} \times T_i + T_{[j,i]} + T_{[i,k+1]}$. These triplets may be computed for all such i, and the minimum chosen to give $T_{[j,k+1]}$. Note that TopProbe([j,k+1]) is also obtained, being that i that returns the minimum triplet. Note that $T_{[j,j+2]}$ simplifies to $p_{[j,j+2]} \times T_{j+1}$ for $0 <= j < n-1$, by proposition 1.

Proposition 4 : Assume that $T_{[c,f]}$ and TopProbe([c,f]), $j <= c < f <= k$ are known for all possible segments [c,f] for fixed j,k. Then $T_{[u,k+1]}$ and TopProbe([u,k+1]) may be successively calculated for decreasing values of u from k to j using the expression in proposition 3.

Proof: By induction on r. Basis step : For $u = k$, the result is trivially true. Induction Step : Let the proposition be true up to $u = i > j+1$. Then for $u = i-1$, the proposition is also true. It follows from proposition 3 that to calculate $T_{[i-1,k+1]}$ the terms required are of the form $T_{[a,b]}$, $i-1 <= a < b <= k$ for all segments [a,b] or of the form $T_{[s,k+1]}$, $i <= s < k+1$, for all values of s. All terms of the first form are available as per statement of proposition 4. All terms of the second form are available as per induction hypothesis. Therefore, the expression in proposition 3 may be used to compute $T_{[i-1,k+1]}$. It follows from proposition 3 that TopProbe([i-1,k+1]) is also obtained.

Proposition 5 : At the end of the procedure embodied in proposition 4, $T_{[c,f]}$ and TopProbe([c,f]), $j <= c < f <= k+1$, will be known for all possible segments [c,f] for fixed j,k.

Proof: From the statement of proposition 4, $T_{[c,f]}$ and TopProbe([c,f]), $j <= c < f <= k$ are assumed known for all segments [c,f]. Therefore, the values that remain to be calculated are $T_{[i,k+1]}$ and TopProbe([i,k+1]) for all values of i, $j <= i < k+1$. The procedure in proposition 4 successively calculates all $T_{[i,k+1]}$ and TopProbe([i,k+1]) for all values of i, $j <= i < k+1$.

Proposition 6 : $T_{[0,i]}$ may be successively calculated for increasing values of i from 2 to n, using the procedure embodied in proposition 4.

Proof: By induction on i. Basis step : For i=2, the result is trivially true, as $T_{[0,1]}$ is known and proposition 4 may be used to compute $T_{[0,2]}$ and TopProbe([0,2]). Induction Step : Let it be true that the procedure in proposition 4 has been used successfully to calculate $T_{[0,i]}$ for all $i <= t < n$. It follows from proposition 5 that $T_{[0,t+1]}$ can be calculated using the procedure in proposition 4.

3.2. The Optimal Algorithm

The optimal algorithm is the procedure described in proposition 6, above. This algorithm has been implemented in C and tested on a micro-vax. An example is provided here as an illustration.

Let $n = 4$, $T_1 = 3$, $T_2 = 5$, $T_3 = 4$, $p_{f_1} = 0.25$, $p_{f_2} = 0.25$, $p_{f_3} = 0.125$, $p_{f_4} = 0.375$.

The solution proceeds by finding $T_{[0,1]}$,$T_{[0,2]}$,$T_{[0,3]}$,$T_{[0,4]}$ in that order. $T_{[0,1]} = 0$. by definition. To find $T_{[0,2]}$, $T_{[1,2]}$ must be found. This is also 0 by definition.

Hence, $T_{[0,2]} = p_{[0,2]} \times T_1 = 0.5 \times 3 = 1.50$ and TopProbe([0,2]) = 1, by applying the simplified formula in proposition 3.

To find $T_{[0,3]}$, first $T_{[2,3]}$ and $T_{[1,3]}$ must be obtained in that order. $T_{[2,3]}$ is 0, and $T_{[1,3]}$ may be found by using proposition 4 to be 1.875, TopProbe([1,3]) = 2. $T_{[0,3]}$ may now be calculated using proposition 4.

$T_{[0,3]} = \text{Minimum}[\ (p_{[0,3]} \times T_1 + T_{[0,1]} + T_{[1,3]}), (p_{[0,3]} \times T_2 + T_{[0,2]} + T_{[2,3]})\ ] = \text{Minimum}[\ (0.625 \times 3 + 0 + 1.875),(0.625 \times 5 + 1.50 + 0)\ ] = \text{Minimum}[3.75,4.75] = 3.75$ and TopProbe([0,3]) = 1.

To find $T_{[0,4]}$, first $T_{[3,4]}$, $T_{[2,4]}$ and $T_{[1,4]}$ must be found in that order. $T_{[3,4]} = 0$. $T_{[2,4]} = 2$, TopProbe([2,4]) = 3, $T_{[1,4]} = 4.875$ and TopProbe([1,4]) = 3.

Using proposition 4, $T_{[0,4]} = \text{Minimum}[\ (T_1 + T_{[0,1]} + T_{[1,4]}), (T_2 + T_{[0,2]} + T_{[2,4]}), (T_3 + T_{[0,3]} + T_{[3,4]})\ ] = \text{Minimum}[7.875,8.5,7.75] = 7.75$ and TopProbe([0,4]) = 3.

The optimal strategy may be obtained by successively indexing TopProbe([l,r]), starting with TopProbe([0,4]). Thus the optimal strategy for this example is -

```
probe 3;
if 3 is OK then
 $f_4$ is faulty;
else {
        probe 1;
        if 1 is Not-OK then
        $f_1$ is faulty;
        else {
                probe 2;
                if 2 is Ok then $f_2$ is faulty;
                else $f_3$ is faulty;
            }
    }
```

4. Extensions, Improvement and Limitations

4.1. Time Complexity

The time complexity of the algorithm is $O(n^3)$. This is evident from the expression for the total number of operations Q. $Q = \sum_{i=2}^{n} \sum_{k=0}^{i-2} \sum_{j=k}^{i-2} (3+1)$. The first summation represents the values of $T_{[0,i]}$ that have to be obtained. The second summation represents the values of $T_{[k,i-2]}$ that must be found before $T_{[0,i]}$ can be calculated. The third summation represents the number of operations to calculate $T_{[k,i-2]}$. The term 3 follows from the expression in proposition 3. Every constituent expression within the expression to be minimized has 3 operations (two additions and one multiplication). The 1 that is added to 3 represents the assumption that choosing the minimum of v elements is equivalent to v operations.

4.2. Extensions

The fact that this algorithm has to be run just once for any system and that a high-level description of a device will not result in large n, implies that an $O(n^3)$ algorithm should be computationally acceptable. Heuristics can also be tested out against this optimal solution for at least up to an n in the high hundreds.

The algorithm can also be applied in hierarchical fashion to optimally localize the fault to some module and then used to find the optimal strategy tree within that module. This is a straightforward extension.

There is another easy extension to troubleshoot the multiple-fault case. The optimal strategy for [0,n] is used repetitively to locate one fault at a time. Since the algorithm identifies a faulty function f_i by finding that node i-1 is OK and node i is Not-OK, it follows that the first function to be identified as faulty will be the faulty function with the minimum i. The component that provides i is replaced, and on checking node n, it is found that at least one fault remains. The search can now be confined to [k+1,n], assuming that the new component is functioning correctly. The optimal strategy for troubleshooting [k+1,n] is also known, as this was calculated by the algorithm in deducing the strategy tree for [0,n] and may now be used. This procedure can be used repetitively until all faults are located.

In many troubleshooting situations, numerical data are not known. The algorithm may be applied to qualitative data by substituting weights for the qualitative values for relative failure frequencies and times of measurement. The p_{f_i}'s thus generated may need to be normalized.

4.3. Limitations

The model imposes restrictions on the systems that this technique may be applied to. Some examples are -

- A pure chain structure is assumed.
- All measurements are bi-valued.
- Compensating failures are not allowed.
- The definition of optimality does not include the run-time of the algorithm.

4.4. Conclusion

The chief contribution of this work is an optimal algorithm to do probe selection in causal chains. A noteworthy feature of the model used is that it assumes that probes may have different costs of measurement. The algorithm runs in polynomial time.

Acknowledgement

We would like to thank Allen Newell and Jay Kadane for helpful discussion and for pointing out interesting paths for future exploration.

References

1. Brule, J.D., R.A. Johnson and E.J. Kletsky. "Diagnosis of Equipment Failures". *IRE Trans. Reliability and Quality Control RQC-9*, 2 (April 1960), 23-34.

2. Hamscher, W. and R. Davis. Issues in Model Based Troubleshooting. Tech. Rept. AI Memo 893, MIT Artificial Intelligence Laboratory, March, 1987.

3. Iwasaki, Y. and H.A. Simon. "Causality in Device Behavior". *Artificial Intelligence 29* (1986), 3-32.

4. Preparata, F.P., G. Metze and R.T. Chien. "On the Connection Assignment Problem of Diagnosable Systems". *IEEE Transactions on Electronic Computers EC-16*, 6 (December 1967), 848-854.

5. Ramamoorthy, C.V. A Structural Theory of Machine Diagnosis. Proceedings of the Spring Joint Computer Conference, 1967, pp. 743-756.

6. Sembugamoorthy, V. and B. Chandrashekran. Functional Representation of Devices and Compilation of Diagnostic Problem-Solving Systems. In Kolodner J., Reisbeck C., Ed., *Experience,Memory and Reasoning*, Lawrence Erlbaum Associates, 1986, pp. 47-73.

7. Simon, H.A. and J.B. Kadane. "Optimal Problem-Solving Search: All-or-None Solutions". *Artificial Intelligence 6* (1975), 235-247.