Learning Approximate Diagnosis

Yousri El Fattah
fattah@ics.uci.edu
Paul O'Rorke
ororke@ics.uci.edu

Technical Report 91–64

September 16, 1991

This research was supported in part by National Science Foundation Grant No. IRI8813048, Douglas Aircraft Company, and the University of California Micro-electronics Innovation and Computer Research Opportunities Program.

Thanks to Pat Langley, Deepak Kulkarni, and other researchers in the AI Research Branch at NASA Ames Research Center for valuable discussions.

Learning Approximate Diagnosis

Yousri El Fattah (fattah@ics.uci.edu)
Paul O'Rorke (ororke@ics.uci.edu)

Department of Information and Computer Science
University of California, Irvine, CA 92717
Phone: (714) 856-6226
Fax: (714) 856-4056

Abstract *Model-based diagnosis (MBD) provides several advantages over experiential rule-based systems. A principal shortcoming of MBD is that MBD learns nothing from any given example. An MBD system facing the same task a second time will incur the same computational effort as that incurred the first time. Our earlier work on incorporating explanation-based learning (EBL) in MBD [4] suggested a diagnostic architecture integrating EBL and MBD components. In this architecture, EBL was used to learn diagnostic rules. But the diagnoses proposed by the rules could be erroneous. So constraint suspension testing was used to check all proposed diagnoses. Insisting on perfect accuracy causes the performance of this scheme for "learning while doing" to deteriorate rapidly with the size of the device to be diagnosed. In this paper, we describe a method for trading off accuracy for efficiency. In this approach, most diagnosis problems are handled by the associational rules learned from previous problems. Model-based reasoning and learning are activated only when performance drops below a given threshold. We present empirical results on circuits of increasing number of components illustrating how this approach scales up.*

AI Topic: Diagnosis, Machine Learning, Model-Based Reasoning.

Domain Area: Diagnosis.

Language/Tool: PROLOG running on Sun 3/60 workstations.

Status: Implemented and tested.

Effort: 1 person year.

Impact: Learning methods provided speedup as compared to diagnosis without learning

1 Introduction

Model-based diagnosis (MBD) provides some definite advantages over experiential rule-based systems. A principal shortcoming of MBD is this. In spite of the fact that MBD performs more complex "first principles" reasoning, it learns nothing from any given task. This means that an MBD system facing the same task a second time will incur the same computational effort as that incurred the first time.

There are various sources problems with earlier MBD methods. Computational schemes have been proposed integrating the probabilistic and logical approaches, incorporating knowledge about how components fail, and mixing the various diagnostic steps in a way that exploits focusing heuristics [1]. These improvements to MBD, in our view, do not substitute for the role of learning. Enhancement of these improvements may still be attained by incorporating a learning component.

Explanation-based learning (EBL) [3, 7], provides a natural framework for learning compatible with MBD. Previous works have considered the use of EBL to speed up model-based diagnosis. See Resnick [5] and Zercher [8]. Those works operate under the *single-fault assumption.*

Our work on the use of EBL for speeding-up MBD [4] is based on Reiter's approach to diagnosis [6]. The goal of this approach is to determine *all* minimal multiple fault candidates that would explain input/output observations. Two learning approaches for incorporating a learning component with MBD were described in [4]. The two approaches differ in the extent of interaction between the learning and the MBD components. In the "learning while doing" approach, MBD assumes a major role. In the "learning in advance" approach, MBD assumes a minimum role. In both approaches learning occurs in two steps of the diagnosis process. The "conflict-set recognition" step is learned dynamically in the "learning while doing" (EBL) approach, but statically in the "learning in advance" (STATIC) approach. Both approaches learn the candidate generation step dynamically. STATIC incurs an extremely expensive computational cost up front, but the cost is amortized when the number of problems reaches a break-even point. Beyond that point, STATIC achieves a dramatic speed-up in comparison with MBD without learning. Learning while doing does not produce positive results in general, due to the fact that it must fall back on MBD to test and debug generated diagnostic candidates. This checking is necessary to ensure that the learning system does not miss diagnostic candidates or generate overly general candidates.

In this paper, we relax the requirement on completeness and specificity of the diagnostic candidates. We allow the learning component to make errors in a training phase where it is given feedback on its actual performance.

We explore a trade-off between efficiency and completeness. Learned (compiled) predictions improve efficiency but they are generally incomplete. This leads to diagnostic hypotheses that are less accurate than those obtained by model-based diagnosis. In many

situations, this loss of accuracy may be a tradeoff worth making in order to achieve improved efficiency.

2 Model-Based Diagnosis

The MBD system uses the theory given by Reiter [6] and emulates the GDE system of de Kleer and Williams [2]. Diagnosis is performed as a 3-step process:

Prediction by propagating observations of premise variables through all constraints

Conflict recognition by determining all (minimal) assumptions responsible for discrepancies between predictions and observations

Candidate Generation by finding all minimal set covers of the collection of conflict sets

The general system architecture is shown in figure 1. The procedure for propagating value inferences is triggered by value assignments of premise variables. We record for each inference an assumption label. Only new value inferences with minimal assumptions are recorded. Conflict set recognition is performed by comparing the predictions with the premise assignment. If there is a discrepancy, then the support set of the inference is declared as a conflict set. For the candidate generation step we implemented Reiter's HS-Tree algorithm [6].

3 Learning While Searching

MBD involves search for: 1) predictions, and 2) minimal candidates. The search for prediction is performed via propagation of premises through constraints in various supporting environments. The search for candidates involves set covering.

Replacing the propagation procedure with associational rules between premises, assumptions and predictions has the following benefits:

1. The problem of finding predictions becomes backtrack-free.
2. Inferences made for all non-output variables, serving only as intermediate inferences are spared.

Replacing the search for set-covering by associations between collections of conflict sets and collections of minimal hit sets eliminates the cost of the HS-Tree algorithm which is exponential in the worst case. However, for the circuits we have empirically studied, the cost of the prediction and conflict-recognition phases dominate the cost of the candidate generation phase.

4 A Problem With EBL for MBD

In standard EBL one learns a sufficient characterization of a concept by generalizing an example instance using a given theory. Multiple-fault diagnosis, as formulated by Reiter [6], requires the knowledge of *all* conflict sets. If we miss some (minimal) conflict sets then the minimal diagnosis will necessarily be incorrect, leading to overgeneral diagnoses.

EBL can be used to learn the predictions as associations with subsets of the premises and the assumptions. This provides sufficient conditions for the prediction but not the necessary conditions needed in MBD. In order to ensure completeness it is necessary to fall back on the model.

Consider the 1-bit full adder shown in figure 2. Given the assignments [1,1,1] for the inputs: [in1, in2, in3], and [0,0] for the output: [out1, out2], the prediction rules (p-rules) learnt for the outputs will be as follows:

$$\neg ab(A1) \wedge \neg ab(O1) \wedge in1 = 1 \wedge in2 = 1 \rightarrow out1 = 1 \quad (1)$$

$$\neg ab(X2) \wedge \neg ab(A2) \wedge \neg ab(O1) \wedge out2 = 0 \wedge in3 = 1 \rightarrow out1 = 1 \quad (2)$$

$$\neg ab(X1) \wedge \neg ab(X2) \wedge in1 = B \wedge in2 = C \wedge In3 = E \wedge xor(B, C, D) \wedge xor(D, E, A) \rightarrow out2 = A \quad (3)$$

Consider the subsequent example where the input is [0,1,1] and the output [0,0]. Obviously rules 2 and 3 both apply, making predictions for both outputs. This is not a complete prediction set of rules, since the following p-rule is applicable to the example, and provides a new assumption label.

$$\neg ab(X1) \wedge \neg ab(A2) \wedge \neg ab(O1) \wedge in1 = A \wedge in3 = 1 \; in2 = B \wedge xor(A, B, 1) \rightarrow out1 = 1 \quad (4)$$

The missing rule will cause the conflict set $\{\neg ab(X1), \neg ab(A2), \neg ab(O1)\}$ to be missed and the diagnosis proposed by the learned rules will be too general.

The approach to combining EBL and MBD that double checks all proposed diagnoses with constraint suspension testing [4] outperforms MBD on small devices but its performance degrades rapidly. As the size of the device increases, the testing quickly overcomes the benefits of EBL caching. In the following sections, we describe a new method that limits this costly double-checking.

5 Learning Predictions

The prediction problem involves finding *all* predictions and their associated minimal support environments for a given premise instance (I/O values). The problem can be solved by searching the entire value inference space until no new inference can be made.

Applying EBL to the prediction problem can be explained as follows.

EBL-Propagate
LOOP:

1. for each constraint
2. for all trigger variables
   (a) Propagate symbolic value inferences, consistent with the premise instance
   (b) Propagate and store with the inference the associated assumptions and premise

Learning prediction rules is a way of allowing the "reuse" of search efforts on previous diagnoses problems. The predictions being made on previous problems may not have been useful for those problems in terms of dicovering conflict sets. But the cached p-rules may be useful for new problems.

Rules learnt from previous examples are not guaranteed to provide a complete set of predictions for a new example. However, falling back on the model and attempting to search for new rules is expensive. A compromise between completeness and efficiency is proposed by a system that learns "approximate diagnosis". We call that system EBL(p). p is a parameter that stipulates a performance threshold. Performance is defined in terms of the percentage of problems where the same set of all minimal candidates as MBD is produced. Note that this is a demanding standard. Even partially correct proposals are counted as incorrect diagnoses. When the actual performance drops below the threshold, learning while making model-based predictions is activated. Otherwise, only learnt p-rules are used for producing the predictions. The EBL(p) algorithm is shown below. Note that EBL is the same as EBL(1).

EBL(p)

1. INITIALIZE: $Learn \leftarrow Yes$, $Problem \leftarrow 1$, $Error \leftarrow 0$
2. LOOP:
   (a) IF $Learn = Yes$ THEN EBL-Propagate ELSE make predictions based on existing p-rules
   (b) Determine all conflict sets
   (c) Determine all minimal candidates
   (d) If the candidates are incorrect then $Error \leftarrow Error + 1$

(e) $Problem \leftarrow Problem + 1$

(f) If $Error < (1 - p) \times Problem$ then $Learn = No$ ELSE $Learn = Yes$

6 Empirical Results

We have carried out an empirical study to compare the performance of EBL(p) and MBD. We studied the performance on the polybox (figure 3) and the N-bit parallel adder (figure 4). For each experiment 100 problems were generated. Each problem has randomly generated premise and fault assignments. Faults for each problem ranges between 1 and 3 with higher probability assigned for single faults. The faults cover the various components at random. For the N-adder, A faulty component is simulated by complementing its normal output. For the polybox, a fault for a multiplier is simulated by subtracting 1 from its normal output, and an adder by adding 1 to its normal output. The inputs for the polybox takes either the value 2 or 3. We fed the same problem to MBD and EBL(p) for p = 0.9, i.e. 90% completeness.

6.1 Polybox

For this circuit one example is sufficient to learn all the prediction rules. EBL(p) will never turn the learning on following the first example. There are 4 rules for the two outputs. Use of those p-rules to make the predictions is much faster than using the model in a constraint propagation mode. Five d-rules will also be learnt, which eliminates the need to fall back on the HS-Tree algorithm. The main savings come, however, from the p-rules.

6.2 N-Bit Adder

We studied the N-bit adder circuit, figure 4, for various values of N. The number of components for N-adder is 5N. We studied the circuit for N=1,2,3.

6.3 N=1

We consider a threshold $p = 0.9$. This means that the learning aims at 90% correctness in terms of a complete characterization of all diagnoses. Following the first example, learning is turned off. EBL(p) produced complete sets of minimal diagnoses for the following 10 examples, thus having its performance at the 100% level. Examples 12, 13 and 14 included faults that could not be detected by EBL(p). Learning is then turned on, where an additional 3 p-rules are learnt from example 15, an additional 3 from example 16, an additional 4 from example 19, an additional 2 from example 22. Performance keeps improving until it reaches the 90% threshold at example 30. Following example 30, the

model is never used and predictions are made using the p-rules. No further errors occur. The total performance is 97 %.

Notice the flatness of the performance curve in the regions where prediction is based only on the p-rules. This occurs for the initial range from 1 to 14 and then for the final range from 30 to 100. The range of problems where EBL-Propagate is active (from 15 to 30) leads to a steeper section of the curve. The slope of the curve for that range is almost identical to MBD.

6.4 N=2

This circuit has 10 components. Following the first example, EBL-Propagate is turned off. The next 2 examples are diagnosed incorrectly, causing EBL-Propagate to be turned on. Learning then continues till problem 20, where performance again reaches the threshold value. In the learning phase the p-rules have increased from 57 to 123. The p-rules are counted for all the variables, not just the outputs (the ones actually used). Again, note the steepness of the performance curve in the learning region. Problems 44, 51, 64, 70, 74, 88 are diagnosed incorrectly by EBL(p). Since the performance remains above the threshold, EBL-Propagate was not called upon. This means that there are indeed more p-rules to be learnt, but they can be sacrificed without affecting efficiency. Note the flatness of the performance curve for problems 20 and up.

Notice that errors by EBL(p) do not necessarily mean missing the correct candidates. Rather, it means missing any minimal conflict set that would have been found by a "complete" MBD, thus resulting in "over general" diagnoses. For example, one problem corresponded to the inputs:[1,0,0,1,1] and the outputs:[0,1,1], generated by simulating an actual fault at component 6. When the problem was fed to MBD the minimal candiates were: [1,8] [3,4,8] [3,5,8] [6] [8,9] [8,10] and when fed to EBL(p) the minimal candidates were: [6] [8]. This counted as an error, although the correct hypotheses [6] is actually generated by EBL(p). In fact, among the 8 diagnoses by EBL(p) counted as as "errors", only 2 missed the hypotheses that actually included the actual fault.

6.5 N=3

6.5.1 Learning at 90% Level

The first example leads to 31 p-rules. The next two examples are diagnosed incorrectly thus activating EBL-Propagate. Learning remains active till problem 20 when performance reaches the threshold. The number of p-rules at that point is 597 (for all variables). Learning is then turned off, but incorrect diagnoses occur for problems 24, reactivating EBL-Propagate. Learning remain active between 25 and 30, where the p-rules increase to 742. Following 30, no further learning takes place. Again, note the flatness of the curve

in the non-learning phase. The "jumps" that occur in that phase are due to occasional problems where many p-rules are fired and lots of non-minimal conflicts are produced. We have not optimized the p-rules, but we will be experimenting with some techniques to cope with that problem.

6.5.2 Learning at 60% Level

When the EBL(p) threshold is lowered to 60%, EBL(p) realizes a significant speed-up for the same 100 problems. In this case learning occurred only for examples 1, 4-7, 12-14, and 17. This means that using only 9% of the examples lead to 72% of the learning.

7 Conclusion

In previous work (reported in [4]) we studied two strategies for explanation-based "learning in advance" (STATIC-MBD) and "learning while doing" (EBL-MBD). STATIC analyzes a device in advance of diagnosing any faults and diagnosis is then purely associational. EBL caches the results of MBD and consults the cache on new problems, checking the predictions of the cache using constraint suspension testing. STATIC incurs learning and MBD costs "up front" while standard EBL incurs these costs incrementally during diagnosis.

Experimental comparisons of these two learning methods with non-learning MBD on small devices indicate that STATIC is preferable to standard EBL. STATIC pays an initial cost for analyzing a device but once the analysis is complete diagnosis is very efficient. Since STATIC is more efficient than MBD on a per problem basis, the costs of the initial learning are repaid after a number of diagnostic problems. Standard EBL outperforms MBD on some devices but in most cases EBL performs worse than MBD and much worse than STATIC because the cost of constraint suspension testing dominates the advantages of caching. However, as the size of the device increases, the initial cost incurred by STATIC and the incremental cost incurred by EBL increase rapidly. STATIC becomes infeasible for large devices, and MBD outperforms EBL because the cost of double checking increases dramatically.

In this paper, we presented a new method relaxing the requirement that the diagnosis system perform with perfect accuracy. In the new method, learned associational rules are used and model-based reasoning and learning are turned off until performance degrades to an unacceptable level. When too many errors have been made, EBL is re-activated.

Experimental results on the new approach on random faults indicate that it alternates between a relatively high cost per problem (incurred when MBD and learning are turned on) and a low cost per problem (incurred by the associational rules). The lower cost tends to dominate the higher cost, so that the new approach outperforms MBD after a number of examples. The lower the required accuracy, the sooner this crossover point occurs.

The new method assumes that its existing efficient associational rules are applicable to new situations, analyzing and learning only when this assumption leads to unacceptable errors. Standard EBL is a special case of this new method, where no errors are acceptable. The new method is preferable to standard EBL in real-world situations where we are willing to tolerate some errors. In realistic situations, observed faults will tend to form clusters in the space of possible faults. Our method takes advantage of this fact to improve efficiency while making acceptable sacrifices in accuracy.

References

[1] J. de Kleer. Focusing on probable diagnosis. In *Proceedings, AAAI-91*, pages 842–848, 1991.

[2] J. de Kleer and B. C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32:97–130, 1987.

[3] G. DeJong. An introduction to explanation-based learning. In H. E. Shrobe, editor, *Exploring Artificial Intelligence*, chapter 2, pages 45–81. Morgan Kaufmann, 1988.

[4] Y. El Fattah and P. O'Rorke. Learning multiple fault diagnosis. In *Proceedings, The Seventh IEEE Conf. on Artificial Intelligence Applications*, pages 235–239, 1991.

[5] P.Resnick. Generalizing on multiple grounds: Performance learning in model-based troubleshooting. Technical Report AI-TR 1052, MIT Artificial Intelligence Laboratory, February 1989.

[6] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[7] S.T. Kedar-Cabelli T.M. Mitchell, R.M. Keller. Explanation-based generalization: A unifying view. *Machine Learning*, 1:47–80, 1986.

[8] K. Zercher. Model-based learning of rules for error diagnosis. In W. Hoeppner, editor, *Proceedings GWAI-88*, pages 196–205. Springer Verlag, Berlin, 1988.

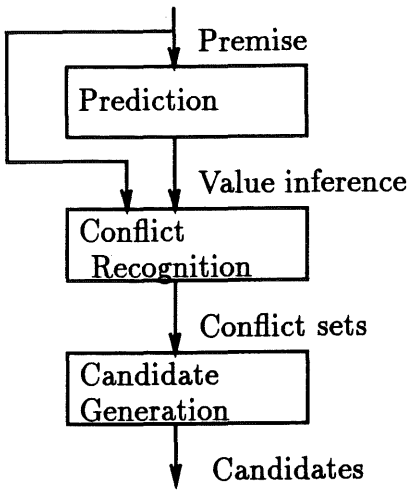

Figure 1: MBD Diagnosis System

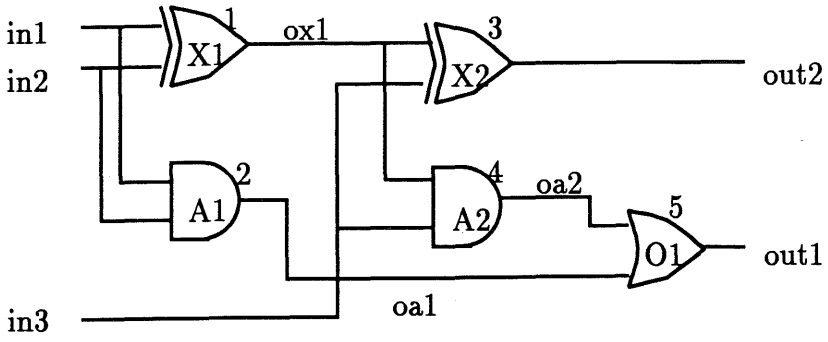

Figure 2: 1-bit Full Adder

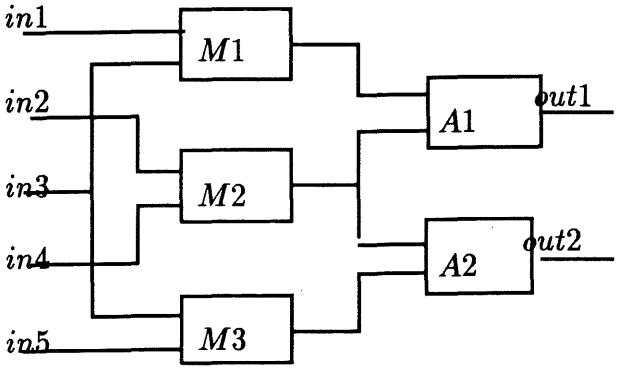

Figure 3: The Polybox Circuit

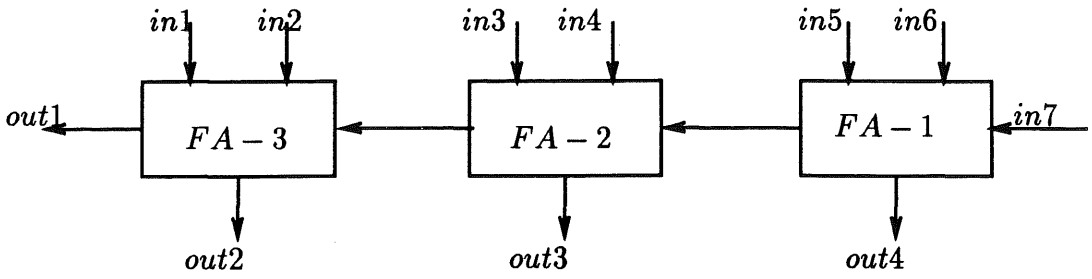

Figure 4: 3-Bit Parallel Adder

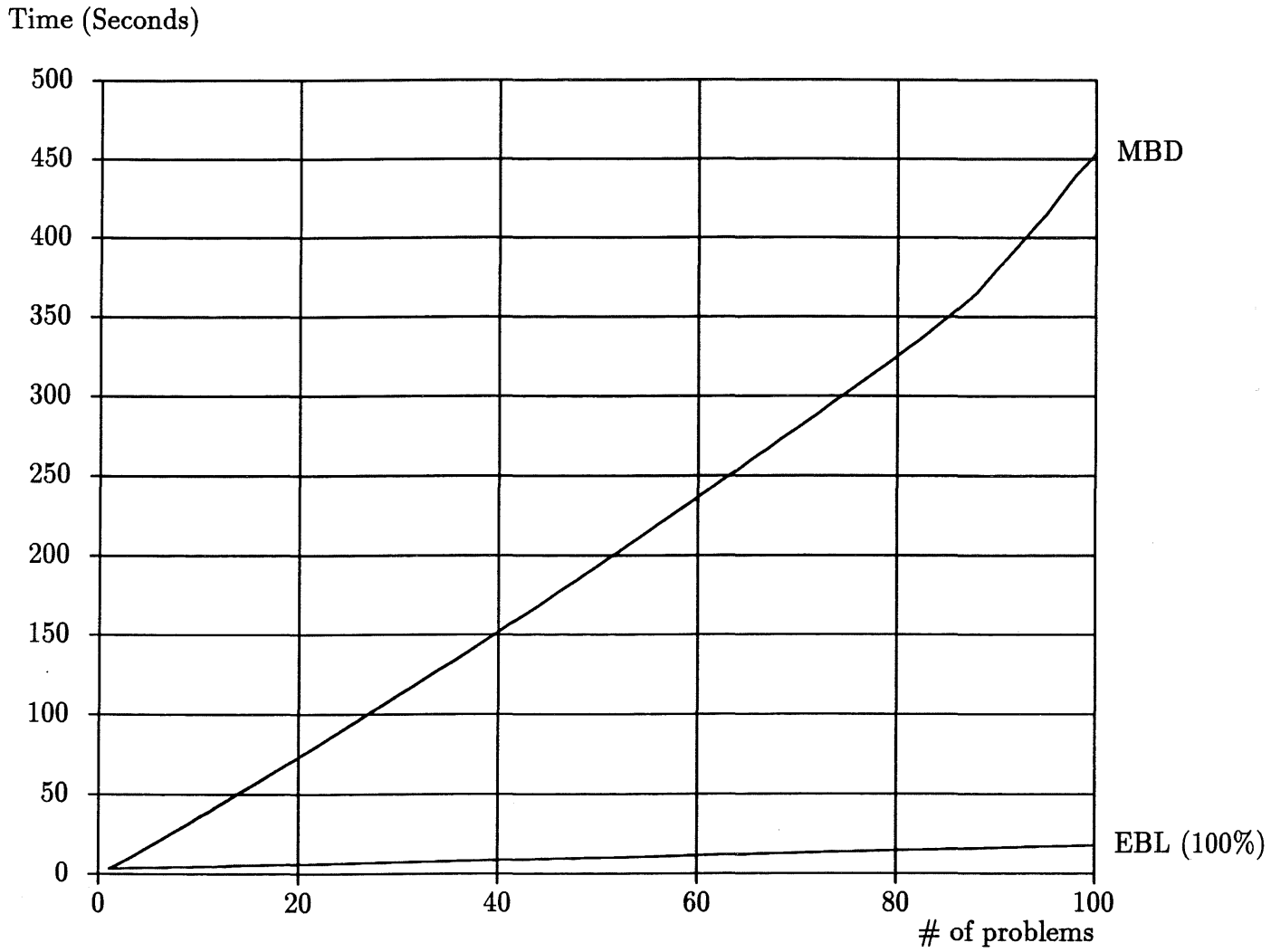

Figure 5: Polybox Empirical Results

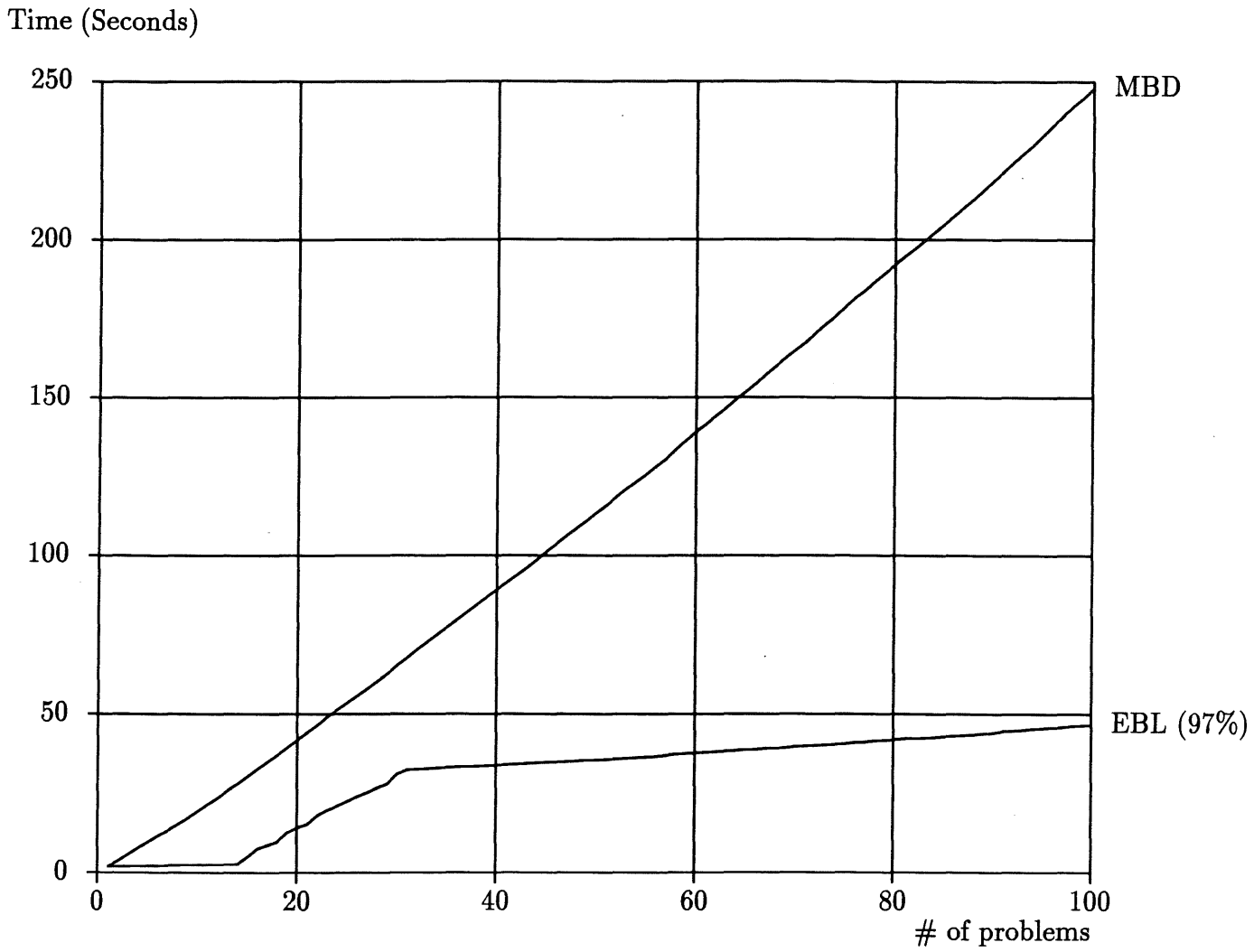

Figure 6: 1-bit adder Empirical Results

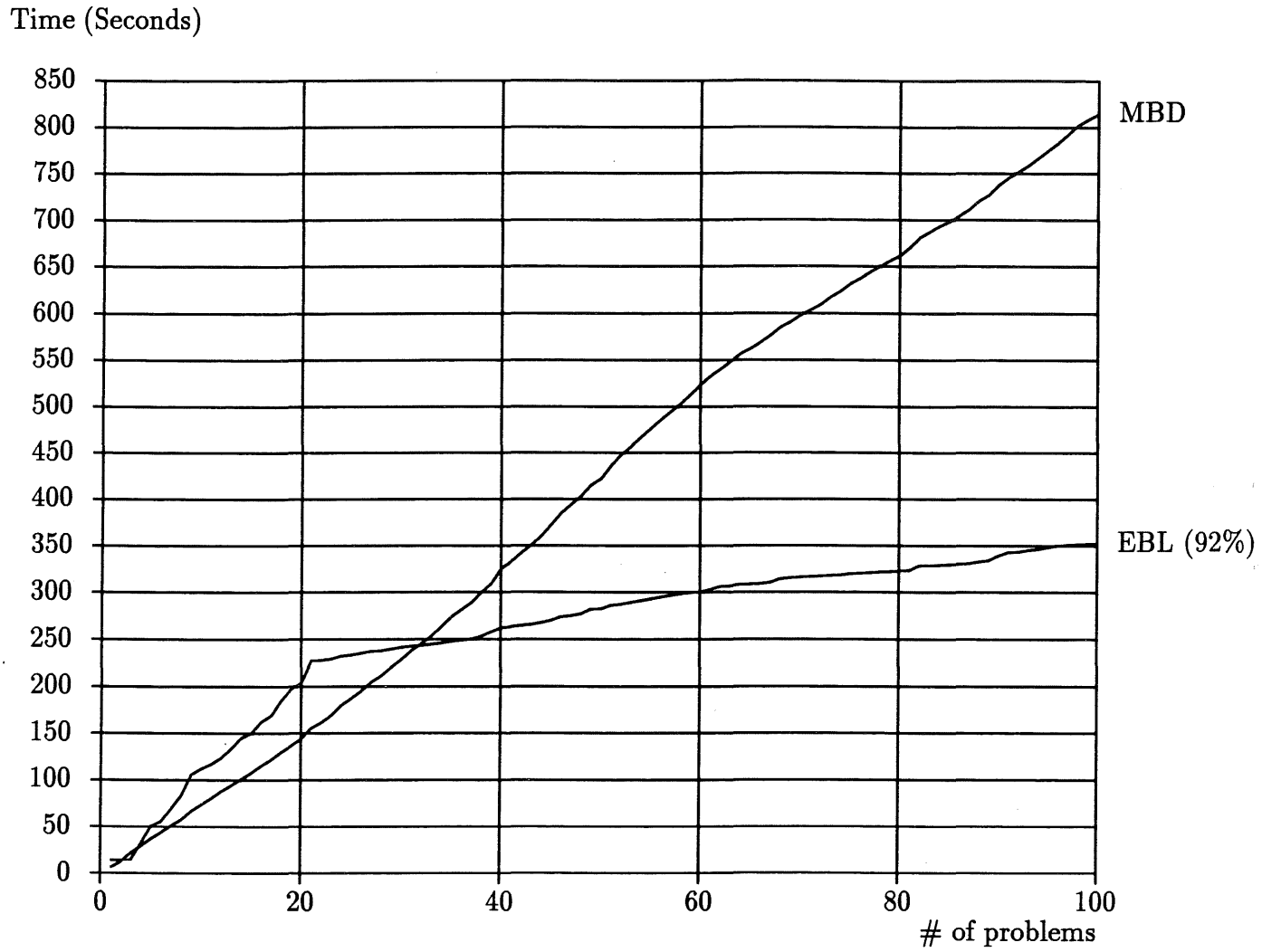

Figure 7: 2-bit adder Empirical Results

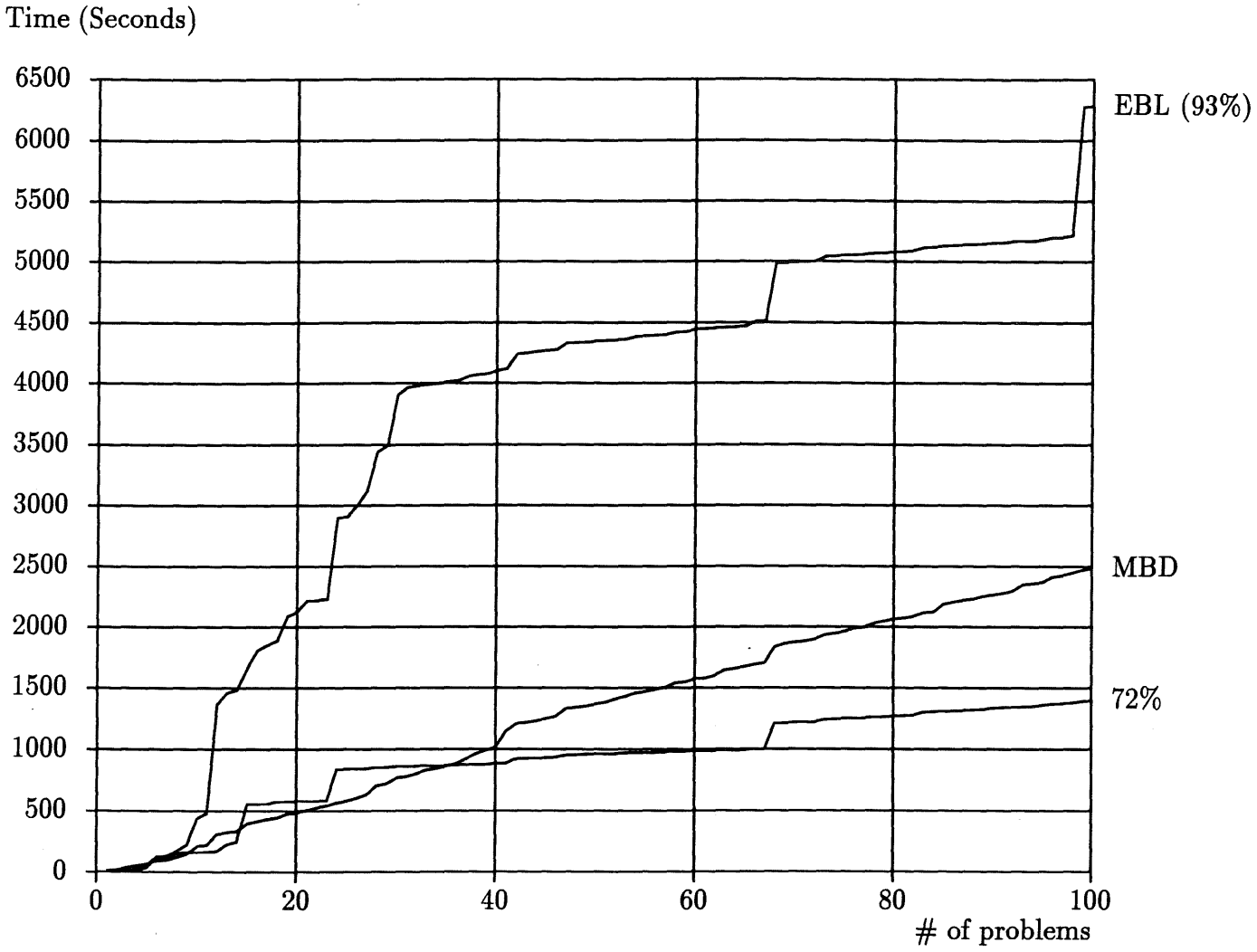

Figure 8: 3-bit adder Empirical Results