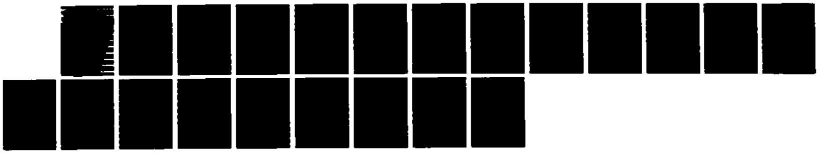
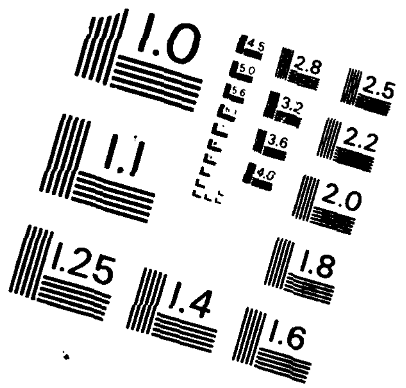
AD-A170 802 SEARCH ALGORITHMS AND THEIR IMPLEMENTATION(U) DUKE UNIV 1/1
DURHAM NC DEPT OF COMPUTER SCIENCE D W LOVELAND AUG 85
AFOSR-TR-86-0530 AFOSR-83-0205

UNCLASSIFIED F/G 9/2 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6

AD-A170 802

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(2)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR- 86-0530	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) ANNUAL REPORT - - AFOSR 83-0205 SEARCH ALGORITHMS AND THEIR IMPLEMENTATION. July 1, 1984 - June 30, 1985		5. TYPE OF REPORT & PERIOD COVERED Annual 7/1/84 - 6/30/85
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) D. W. LOVELAND		8. CONTRACT OR GRANT NUMBER(s) AFOSR 83- 0205
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computer Science Department Duke University Durham, NC 27706		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 2304 A7
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE August 1985
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Air Force Office of Scientific Research Air Force Systems Command Bolling AFB, Washington, DC 20332		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Annual Report

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Papers completed this year include (1) correcting natural language input using expectations, (2) fast algorithms for finding some boundary sets of binary monotone set functions, and (3) a review of automatic programming techniques. Work on search with limited resources and a study of automating rule strength determination for rule-based systems should be completed this coming year. Work continues on approximation algorithms for the test-and-treatment problem and a new effort is underway in learning mechanisms with a focus on a method for comparing learning mechanisms that has already yielded a promising new learning strategy.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclass

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AFOSR-TR- 86-0530

ABSTRACT

Papers completed this year include (1) correcting natural language input using expectations, (2) fast algorithms for finding some boundary sets of binary monotone set functions, and (3) a review of automatic programming techniques. Work on search with limited resources and a study of automating rule strength determination for rule-based systems should be completed this coming year. Work continues on approximation algorithms for the test-and-treatment problem and a new effort is underway in learning mechanisms with a focus on a method for comparing learning mechanisms that has already yielded a promising new learning strategy.

86 8 13 060

ANNUAL REPORT

Search Algorithms and Their Implementation

(AFOSR-83-0205)

July 1, 1984-June 30, 1985

Overview

This is the second year of support under AFOSR-83-0205. The overall goal of the research of this grant is to study concrete uses of and abstract properties of search, the control organization so prevalent in AI applications. Our global approach is summarized in the *Research Objectives* subsection below.

Work this past year primarily focused on work undertaken earlier although in some cases the progress was much more pronounced this year. In particular, this was the case for studies in knowledge-based (expert) systems where we have undertaken the automation of certainty factor definition in rule-based systems. That is, we are interested in understanding when it is feasible to have the value of a rule determined automatically by some calculation procedure, using a set of tests as the data for the calculation. This year we learned much about conditions under which such determinations can be made easily and when it is "provably" hard to compute these values quickly. Some of the algorithms studied are not iterative ("learning") algorithms where one hopes to converge by increments to the right values. There are domains such as the calculus used by the well-known expert system MYCIN where one directly solves (nonlinear) equations by back substitution as part of the solution technique (in the "complete test" case). We expect this work to be successfully concluded this next year with the completion of the doctoral thesis of Marco Valtorta. (Although Marco was a teaching assistant this year and thus not directly supported by this grant he was so supported in the previous years by the Air Force grant and this work is very much a product of this grant.)

We remark that this study is part of an ongoing interest in the area we call *knowledge evaluation* meaning the development of tools of analysis and manipulation regarding the quality of a knowledge base. We see the problem of maintaining the quality of a knowledge base as important as and eventually as difficult as determining the correctness of a program. The above is very pertinent because if rule strengths can be automatically (and correctly) determined then the rule base will be sound.

Another effort that is reaching maturity (in the component now under investigation) is the work on search with limited resources. This investigation is concerned with strategies for search when limited resources constrain the number of points where information can be collected. The work of David Mutchler, almost certain to yield a Ph.D. thesis this year, concentrates on one-person search models where the information points are fewer than needed for even one complete path exploration, let alone checking some alternate paths completely. Thus probabilistic reasoning is heavily involved in this study. Some very interesting results are now known; a straightforward fast approach to the solution is not always correct, but is provably best for a substantial class of problems. It also appears that this "greedy" strategy is a very good approximation to the optimal strategy in the other cases but this is at present only conjecture supported by a moderately large set of experimental evidence. (On the basis of a talk on this project at the Naval Research Labs, David has been invited to join their research group studying applications of information theory to search.)

In another study, we have shown how to implement large semantic nets (with possibly hundreds of thousands of transitions) on a massively parallel computer which is being locally designed and constructed. This system will make it possible to carry out complicated queries on large AI knowledge bases in relatively few machine cycles with the cost of processing being independent of the size of the knowledge base.

In addition to studies reaching a conclusion there are studies in midstream. The problem of designing optimal, or near optimal, test procedures, studied by many researchers, has been extended by us to consider the interaction of tests and treatments. That is, the model(s) we consider recognizes that treatments are often introduced before the precise fault is uncovered. Although the problem is of clear importance nothing has been found in the literature dealing with this extension to the testing problem. One reason may be the difficulty of working with the more complex models. This difficulty may limit what can be analyzed but even basic results such as the gross value of "naive" procedures are missing from the literature. Standard techniques can be used to solve the problem exactly (and we study a parallel algorithm for doing this, as discussed below) but all known algorithms for exact solution are very computationally expensive. We seek good approximations that can be computed quickly. We have made some progress, as we report in the section on test and treatment procedures, but not enough for publication yet. We feel that the results we are approaching are important and this makes it worthwhile to stick with the problem in spite of its difficulties.

A paper reporting a manner of determining optimal test and treatment procedures is nearly completed and will be submitted for publication. This study uses the Boolean Vector Machine, a massively parallel computer which is being locally designed and constructed, and for which a simulator exists. As mentioned above, algorithms are known that are capable of solving the test and treatment problem, but it is also known that they must take a tremendous amount of computing resources. This means an inordinate amount of computing time on a sequential machine, so it is of interest to investigate the speed-up realized on a massively parallel machine. The problem is primarily how to take advantage of the parallelism available in the machine, but a precise formulation of the test and treatment problem with a (costly but) precise solution method also occurs in the paper.

Work on learning has picked up from a quiescent state and some exciting progress is being made in the attempt to give a means of comparison to various learning "machines". The exciting part is a new type of learning machine (strategy) that seems to perform in a near-optimal way.

We have included a section on the development of critical set algorithms, although it was not originally a study goal. This work came out of work in knowledge evaluation but had obvious value of its own and absorbed significant effort this year, effort which seems appropriate given its worth independent of the effort that spawned it. The paper has been submitted for publication.

Research Objectives

Our global concern is to understand better the characteristics and methodologies of search. We realize these aims both by studying search techniques in the abstract (e.g., see [4]) and by applying analytical tools to situations in AI where search is an important component of the solution. For example, in determining attenuations in rule-based inexact reasoning systems, we have invoked quadratic programming methods and tools of algorithm analysis to a domain where "standard" AI technology would use an incremental learning algorithm, which is a weak analytical structure employing much search. As another example, we look for quickly computed very good approximation algorithms for the test-and-treatment problem where a (branch-and-bound) search can give the optimal solution, but at great computational cost. Thus we take on problems important in their own right and seek more effective solutions than "naive" search yields. Besides solving important problems, we expect that further general principles employing analytic methods for naive search will emerge. There is much attention to AI now and much attention to immediate applications, but we are one of the very few places trying to use known methods (or invent our own, such as the critical set algorithm) to improve on standard AI search techniques, i.e. apply good scientific inquiry techniques to search problems.

Research Status

The status report is divided into projects with the personnel involved named in parentheses. Graduate students' names appear before their adviser's name when the adviser's role is primarily problem formation and research guidance.

Expert systems: computing certainty factors (Valtorta, Loveland)

For several years we have had an interest in an area we call *knowledge evaluation* which is concerned with finding ways of assisting the user in maintaining and judging the quality of a knowledge base, presumably a large knowledge base in existence for a number of years. One question we became interested in along this line is determining or adjusting certainty factor weights automatically, so as to make a rule-based system more accurately perform its task. Although the ultimate ideal is to have the system learn the entire rule space from tests (examples) it is a much more modest endeavor to have the rule weight (the "degree of validity or worth" of the rule) established by testing on known cases. It is a natural intermediate step because experts may well know a particular rule-of-thumb but be very unsure how much weight to give the rule outcome in interaction with the other rules of the system. In particular, correlations between rules are very hard to judge; a rule may seem very much worth adding in isolation but in fact nearly be covered by other rules already in the system.

Our approach for this study is different than that of usual learning algorithms. We also are interested in algorithms that would allow us to compute directly the certainty factors, which we call *attenuations* because the strength of the inference made by the rule is attenuated by the amount specified. (A weight of 1 would mean no attenuation.) That is, we wish to understand how difficult it is, and find algorithms where appropriate, to use the information from a block of tests *in any fashion*, not just incrementally, to determine rule strength weights, or attenuations. Perhaps surprisingly, we find that such determinations can be easy in certain cases, although in many cases it is provably difficult almost certainly (i.e., NP-hard) to compute such attenuations.

The models of learning for which there are results are the *complete test case* and the *incomplete test case*. For the complete test case we assume a perfect test environment where any desired test can be performed. The results that we have obtained for the complete case can be summarized as follows:

(a) It appears easy to synthesize attenuations for *trees* with MIN for OR nodes, MAX for AND nodes and multiplicative attenuations. In particular, it is easy when reals are involved (some qualification is needed here) and we conjecture that it is easy for finite arithmetic when rounding is used.

(b) It is hard to synthesize attenuations for *trees* with MIN, MAX and multiplicative attenuations when the predicate functions are allowed to perform "sign flipping" of the certainty factors. (This is performed in MYCIN by the predicate function THOUGHTNOT.)

(c) It is hard to synthesize internal attenuations for *acyclic graphs* using MIN, MAX and multiplicative attenuations.

(d) It is easy to synthesize input attenuations for the same model as in (c).

(e) It is hard to synthesize attenuations for *chains* using MIN, MAX and some choic of attenuations that are not closed under composition.

(f) Similar results seem to hold for probabilistic addition in place of MAX. In particular, it appears easy to synthesize attenuations for trees with real multiplicative attenuations.

In the incomplete test case, a collection of tests is given and the attenuations must be synthesized in such a way that the tests are correctly executed. The results that we have obtained can be summarized as follows:
It is hard to synthesize attenuations for *trees* with MIN, MAX and multiplicative attenuations.

Current work is proceeding on these fronts regarding the incomplete test case:

(a) Synthesis of attenuations even for trees with MIN, MAX, multiplicative attenuations, bounded fan-in to the MIN's, and fixed depth of the tree. Note that this models well some typical rule-based expert systems: bounded number of premises to each rule (independent of the size of the rule base) and short inference chains, also independent of the size of the rule base. In particular, if determining attenuations is hard for this highly constrained model, then finding certainty factors for expert systems using MIN and MAX surely is hard, if done from test cases. At this point we are very close to a proof that it is hard to determine attenuations in this restricted setting.

(b) As above, with probabilistic addition in place of MAX.

(c) Approximate algorithms. We are trying to exploit *the concept of influential input* for the MIN/MAX case with multiplicative attenuations. In this case, the output of the inference net is always equal to the attenuated value of one of the inputs. We have proved that the problem of setting attenuations remains hard for trees even when an influential input is specified for each test.

Search with limited resources (Mutchler, Loveland)

We have been studying "search with limited resources" using a probabilistic model with parameter p $(0 < p < 1)$. We have proved several results this year within this model, most notably as follows.

(1) We had shown last year that the "obvious" decision (a "greedy" decision) is not always the optimal decision. We have extended this result to show that this anomaly can occur *no matter how many node expansions remain to be made.*

(2) We had shown last year that the greedy policy is optimal (ie., that this anomaly cannot occur) when p is a unit fraction. We can now show that the greedy policy is optimal if $p < .59$. The proof of this result incorporates the previous proof; a key lemma invokes the theory of branching processes.

(3) In our model the searcher learns information about *all* arcs immediately below the explored node. It was conjectured that this might be the source of the anomaly [Shimon Even, private (and casual) communication]. We have shown that such is not the case. We can derive similar anomalies from the model wherein the searcher learns information about only a single arc below the explored node.

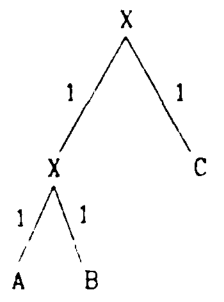
We summarize the problem(s) we have labeled "search with limited resources". The following questions provide motivation for studying search with limited resources:

Where can search resources best be expended to gain "useful" information?
How should one use acquired information to gain more information?
Does knowing the limits of search resources help one search more effectively?
What algorithms are good approximations to the optimal search policy?

To investigate these questions in an analytic framework, we use the following probabilistic model. We generate trees as Karp and Pearl [2] and others have done. That is, we consider complete binary depth N trees. Arcs are independently assigned 1 or 0 with probability p and $q = 1 - p$ respectively. Each leaf value is the sum of the arc values on the path from the root to the leaf. Hence values of sibling leaves are positively correlated. One argument for using the arc-sum model is that real-world search trees often possess this correlation property.

This model features a scout who searches such trees as Karp and Pearl [2] did. The scout begins at the root and *expands* nodes, thereby finding the values of the two arcs below it. No node can be expanded until its parent has been expanded. After expanding n nodes, the scout must halt and report to the general. The general uses the scout's findings to select a leaf of the tree. The general's goal is to minimise the expected value of the chosen leaf. The questions are: what policy should the scout use to select nodes for his n node expansions, and how should the general choose his leaf node?

For example, suppose the scout is searching the depth 5 tree whose top two levels look as pictured below after two node expansions. (Nodes A, B and C are the roots of unexpanded subtrees. The numbers beside arcs indicate expanded nodes.) Let $p = 0.85$. Suppose the scout has resources left for only *two* more node expansions. Which node should he expand first?



Or suppose the pictured tree is the situation that the (exhausted) scout reports to the general. What leaf should the general choose?

The answer to the general's dilemma is not hard. He should choose any leaf below the frontier node with least ***zero value***, where the zero value of frontier node β is

$$\text{sum of arcs on the path from the root to } \beta$$
$$+ \ (\text{distance from } \beta \text{ to a leaf below it}) \times p \, .$$

In the tree pictured above, the general should choose any leaf below node C, whose zero-value is 4.4.

One might then wonder whether the scout should make his decisions based on the same criterion. The answer is both a reassuring "yes" and a surprising "no". Under the condition that n (number of node expansions allowed) is no more than N (depth of the tree), we have shown that

(1) If $p < .59$, this "greedy policy" (expand any node with least zero-value) is optimal.

(2) For any n, if p is large enough, situations may naturally arise wherein applying the greedy policy when n node expansions remain is ***not*** an optimal decision. The tree pictured above is one such situation. The scout should expand either node A or node B!

We have shown that the above problem can be described as a Markov decision process [3]. The proof of the first result is an induction that invokes the theory of branching processes [1] to prove a key lemma.

We have also investigated a variant of this model wherein the scout expands arcs rather than nodes, thereby learning the value of the expanded arc. We have shown how to transform this variant to a Markov decision process that is essentially the same as that for the problem described above. The analysis proceeds similarly and yields corresponding results.

We are currently seeking an answer to the question

when the greedy policy is not optimal, just how bad is it?

The instances studied so far suggest two conjectures which we are now trying to prove:

(1) The optimal decision is never more than "one away" from the greedy policy. That is, it is never optimal to expand a node whose zero-value is neither the smallest nor next-to-smallest of the frontier nodes.

(2) For fixed p, there is an M such that if more than M node expansions remain, the greedy decision is optimal. Hence, the difference between the greedy and optimal policies does not grow with the number of node expansions allowed.

[1] Harris, Theodore E. *The Theory of Branching Processes* (Springer-Verlag, Berlin, 1963).

[2] Karp, R.M. and Pearl, J. Searching for an optimal path in a tree with random costs. *Artificial Intelligence 21* (1983), 99-116.

[3] Ross, Sheldon M. *Introduction to Stochastic Dynamic Programming* (Academic Press, New York, 1983).

Associative Networks on a Massively Parallel Computer (Jackoway, Biermann, Wagner)

Many artificial intelligence projects in the past fifteen years have used semantic networks as their underlying knowledge representation. In a separate realm, recent breakthroughs in very large scale integration have lead to designs for machines with vast numbers of processors. In this study, we have joined these two technologies. A methodology has been found for representing semantic networks onto a massively parallel processor which is currently under development at Duke. The results show:

(1) The time required to process a query is dependent strictly on the pattern of the query, not on the size of the classes being processed. A system built using this knowledge representation will give consistent semantic processing performance.

(2) The order of processing a query does not affect the speed. Thus, there is no need for heuristics and monitors to determine the most efficient way to process a query.

(3) Although we do not receive anywhere near an n-fold speedup by using n processors, we still receive significant performance benefits over a single processor.

(4) The semantic network may be used not just for fact retrieval; it also allows some problems involving numerical minimizations to be solved efficiently.

This work has appeared in an A.M. thesis finished this year. (While the author was not funded directly by this grant because he arrived with support, Profs. Biermann and Wagner were so supported while guiding this work.) We have made some attempt to interest a publisher in the entire thesis as an "lecture notes" type publication, but this process has lagged and now we are encouraging Jackoway to publish a journal article on this work.

Test and treatment procedures (Loveland, Lanzkron)

Work this past year on the test and treatment problem deepened our understanding on how much more difficult this problem is than the binary testing problem from which it derived. (The test and treatment problem is an NP-hard problem as is the binary testing problem.) The Huffman algorithm, which gives an optimal solution to the restricted complete test case for binary tests, can at best serve as a reasonable approximation algorithm in the test and treatment case (under appropriate modification to adapt to the problem at hand.) The trees have compound element nodes and elements can migrate from one node to other nodes and yield improved cost. The rules of migration are still unclear and it remains open as to whether migration algorithms alone will yield optimal trees. Initial experiments suggest that the approximation obtained is very good however. Our priority task now is to find bounds for this apparently very good approximation. A major difficulty to analytical work is finding any reasonable characterization of the optimal trees to get the quality of approximation result we feel should hold

We now give a fuller description of the test and treatment problem.

The test and treatment problem requests the selection of a minimum (or near-minimum) test and treatment procedure under an expected cost criteria. The problem arises whenever a fault (disease, system malfunction) must be treated. The classic example is medical diagnosis and treatment, but other applications also are important, such as computer system fault location and correction and logistical system breakdown correction. In general, the problem exists whenever a sizable population of complex objects (people, ships, computers) must be maintained at reasonable cost.

The problem specification consists of a universe $U = \{0,1,...,k-1\}$ of k objects, each with an associated weight p_i, and a set of tests and treatments T_i, $1 \leq i \leq n$, each with an associated cost. The T_i, $1 \leq i \leq m$, denote tests, and the T_i, $m < i \leq n$, denote treatments. We assume that only one object is actually faulty, its identity is unknown, and each object i has an *a priori* likelihood p_i of being the faulty object.

Each test and treatment is specified by a subset of the universe; if the unknown object is in the test or treatment set then the test responds positively, or "is successful", or the treatment is successful. If the test is successful, one eliminates the other objects from consideration (and if negative, one eliminates the test set of objects), while a successful treatment ends the procedure. A failed treatment means the processing must continue. Cost of a procedure is an expected cost; i.e.,

$$\sum_i (cost\ of\ detecting\ object\ i)(p_i)$$

Because nothing is known about this class of problems (at least, judging from a literature search), we have had to formulate the problem before attempting a solution. The general problem is formulated in a paper nearing completion by R. Wagner, Y. Han, D. Duval and D. Loveland where a dynamic programming technique is used to solve the problem. The technique is classical so the interest in this paper lies primarily in the parallel implementation of the particular dynamic programming formulation.

However this general solution has exponential running time in the number of objects and this is almost surely the case for all general solutions (the problem is NP-hard), although parallel processing helps greatly here. We have therefore also focussed on fast (polynomial) algorithms for special cases and approximation algorithms for the general case. Let us call the case where all tests and treatments are available the *complete TT problem*. We have had hopes that the complete TT problem (with certain assumptions regarding costs) has a nice algorithm for selecting the minimal cost TT procedure because the well-known Huffman algorithm is an elegant fast algorithm for the related binary testing problem. We now see that interesting anomalies are arising that seem to thwart a clean algorithm formulation. These are the migration of elements from one node to another, where a node represents a set of treatments. Whether we can find a pattern to this migration effect that yields a reasonable optimal algorithm or whether we settle for a good approximation algorithm is still unclear. At present the pattern of migration seems complex enough to suggest that it may not be worth knowing the optimal algorithm (because it would be too complex for use) and attention to approximation algorithms is a more productive route.

Errors Made by Learning Machines (Biermann, Gilbert, Nigrin)

Learning has been an interest of ours for many years but the project we report on here got its start this year with the realization that no method for comparing learning procedures was available. In trying to understand how such comparison might be handled we have in fact discovered a new type of learning machine that seems to give near optimal behavior, as we mention below.

We examine learning machines which can acquire binary-valued functions f of binary valued variables $x_1, x_2, \ldots, x_m$ on the basis of example behaviors. As an illustration, such a learning machine might be presented examples of feature values for class A and be expected to self-adapt to properly recognize all members of class A and to reject all nonmembers. Such learning machines have been studied in the past by Minsky and Papert (perceptrons), Samuel (signature tables), Valiant (conjunctive and disjunctive normal form systems), and many others.

We examine, in particular, the level of expected errors made and the rate of learning by such machines as they relate to the number of functions they can acquire. Thus a learning machine which can acquire only a few behaviors (or functions) can learn very quickly because only a few bits of information need to be gathered from the examples. However, a learning machine which can acquire almost any behavior will need many bits of information to select which behavior is to be acquired.

A learning machine is characterized partly by the number of functions it can learn and partly by the distribution of those functions over the space of all possible functions. That is, suppose a machine is to learn one function on four variables out of the space of $2^{2^4} = 65K$ possible functions, but it is capable of learning only one of 520 different functions. Then if the machine is required to learn some function not in its small repertoire, the best it can do is to select a learnable function which is near the target function. It should acquire the behavior that agrees with the target behavior on as many bits as possible. The question is, then, how many errors will the learned function make on the $2^4 = 16$ possible inputs? We find that if the 520 learnable vectors are optimally distributed to minimize expected error, the expected error will be about *2* bits out of 16. If the 520 learnable vectors are distributed in the worst possible way to maximize expected error, the expected error will be about *5*. Knowing these two limits, we can then compute the expected error for any learning machine, compare it with the known bounds and make a judgement on whether it has good performance. For example, a signature table of the kind used by Arthur Samuel which has 520 learnable vectors has an expected error of 2.56, indicating its learnable behaviors are relatively well distributed over the space of functions. However, we have found that perceptron learnable behaviors seem to be not so well distributed and yield higher errors.

These considerations lead to the concept of the *optimal learning machine* which has its learnable functions spread across the function space so as to minimize expected error. It turns out that such machines are extremely difficult to construct. By using enumerative methods, we have constructed many of them and have studied their properties carefully. The major result of this work has been the discovery of a new type of learning machine, called the G-machine, which closely approximates optimal behavior and which is relatively inexpensive to construct. Current efforts are aimed at improving the characteristics of the G-machine and understanding its properties.

In summary, this series of studies attempts to build tools for understanding quantitatively a large class of learning machines and for building new ones. The primary parameters of interest are the number of learnable behaviors, their distribution over the function space, and their associated rates of error and rates of learning. The tools are yielding considerable information regarding well known learning machines and they lead to methodologies for building new ones.

Critical sets (Loveland, Lanzkron)

Our previous work on knowledge evaluation had led to the discovery of a useful algorithm that surprisingly does not seem to have appeared in the literature. This was ascertained not only by a brief literature search but by asking those in the field most likely to know. Although none knew of any previous occurrence of the algorithm in the literature, all expressed surprise that it seemed not to be there. Several even contributed interesting approaches to the problem, including the referee of our first submission in this regard. With these new ideas in hand we discovered that there were several approaches to the problem, and that our original discovery appeared to remain the most attractive but there were close runner-up algorithms. Overall, the clarity of the algorithms was greatly improved by this second go-around. We have resubmitted the paper for publication.

A critical set is defined in terms of a binary monotone set function, that is, a function f defined over all subsets of a universe U taking only values 0 and 1 and satisfying $f(S)=0$ if $f(T)=0$ and $S \subseteq T$. A set S is critical iff $f(T)=1$ for all T such that $S \subseteq T$ and $f(R)=0$ for all R such that $R \subset S$. Upper bounds on the worst case times for the algorithms, determined by the number of calls to the binary set function, vary downwards from $2r\lceil lg_2\, n\rceil$ function calls. Here n is the size of U and r is the size of the critical set found. We find that the conceptually easiest algorithm is not the easiest to program and also that the algorithms with better upper bounds fare more poorly in practice, based on a small sample of random trials. Although finding one critical set is easy we also show that it can quickly get very difficult to find additional critical sets, simply because it can be hard to separate known critical sets from further possible critical sets.

The initial application occurred in a proposed method for aiding the expert to reduce the number of tests needed to check induced ambiguities when a new inference rule was introduced into a classification-type rule-based system. (The application was reported in the paper *Detecting ambiguities; an example of knowledge evaluation* given at the 1983 International Joint Conference of AI.) Since then a new application of critical sets has been found by Mr. Valtorta in work on computing attenuations. (See the section *Expert systems: computing certainty factors;* the computation of input attenuations for acyclic graphs using max and min with multiplicative attenuation uses critical sets.) The bottom line is that the critical set algorithms are likely to be useful in a number of settings.

Boolean Vector Machine (Wagner, Han)

The past year of partial support has been devoted to revising our joint paper on the test-and treatment problem. Additional effort has been devoted to identification and correction of problems in the hardware (VLSI chip) being constructed for the machine itself, and to preparation of a proposal for future research, submitted last November.

Progress in these areas has been substantial -- two months of diagnostic testing (mostly funded by NSF) identified a problem in our chip's memory design, and resulted in a new submission to MOSIS. That new version, a slight modification in our previous design, is currently under test, and appears to have solved the problems our chip's memory had. The specific problem, an inability to retain stored data for longer than 200 μsec, would have made the chip nearly unusable in a large system. We hope to obtain enough of these chips to construct a prototype BVM of 2048 PE's (32 chips), a machine large enough to test our algorithms at full speed.

Supported Personnel

Loveland, Donald W. (Principal Investigator)

Biermann, Alan W. (Co-principal Investigator)

Wagner, Robert A. (Co-principal Investigator)

Lanzkron, Paul (Research Assistant)

Mutchler, David (Research Assistant)

Nigrin, Albert (Research Assistant)

Valtorta, Marco (Research Assistant; summer only)

Publications and Reports

July, 1983 - June, 1985

Chronologically ordered

1. Biermann, A. Dealing with search. *Automatic Program Construction Techniques* (Eds. Biermann, Guiho, Kodratoff). MacMillan Publ. Co., 1984.

2. Ballard, B. The *-minimax search procedure for trees containing chance nodes. *Artif. Intelligence,* Oct. 1983, pp. 327-350.

3. Loveland, D. Performance bounds for binary testing with arbitrary weights. *Acta Informatica* 22(1985), pp.101-114.

4. Ballard, B. Non-minimax search strategies for minimax trees: theoretical foundations and empirical studies. Duke C.S. report CS-1983-13, July, 1983. Conditionally accepted in *Artif. Intelligence*.

5. Reibman, A., B. Ballard. Non-minimax search strategies for use against fallible opponents. *Third Natl. Conf. on Artif. Intell.*, Washington, D.C., August, 1983. Conditionally accepted in *Artif. Intelligence*.

6. Fink, P. The acquisition and use of dialogue expectation in speech recognition. Ph.D. Thesis, Computer Science Department, Duke University, 1983. Also C.S. Report CS-1983-101.

7. Jackoway, G. Associative networks on a massively parallel computer. A.M. Thesis, Computer Science Department, Duke University, 1984.

8. Ballard, B., N. Tinkham. A phrase-structured grammatical framework for transportable natural language processors. *Amer. J. Comput. Linguistics,* 1985, pp. 81-96.

9. Valtorta, M. A result on the computational complexity of heuristic estimates for the A* algorithm. *Infor. Sciences 34*, 1984, pp. 47-59.

10. Biermann, A.W. Automatic programming: a tutorial on formal methodologies. To appear in *Jour. of Symbolic Comp.*

11. Fink, P.K., A.W. Biermann. The correction of ill-formed input using history based expectation with applications. Accepted in *Computational Linguistics,* 1985.

12. Loveland, D.W. Finding critical sets. Duke C.S. Report CS-1985-22, submitted for publication.

Note: Bruce Ballard, while not funded by this grant this year, was funded last year, and in previous years on an earlier AFOSR grant.

Conference Presentations

July, 1983 - July, 1985

Chronologically ordered

1. Reibman, A.L. and B.W. Ballard. Non-minimax search strategies for use against fallible opponents. *Third Natl. Conf. on Artif. Intell.*, Washington, D.C., August, 1983.

2. Loveland, D.W. and M. Valtorta. Detecting ambiguity: an example of knowledge evaluation. *Eighth Intern. Joint Conf. on Artif. Intell.*, Karlsruhe, W. Germany, August, 1983.

3. Valtorta, M. A result on the computational complexity of heuristics for the A* algorithm. *Eighth Intern. Joint Conf. on Artif. Intell.*, Karlsruhe, W. Germany, August, 1983.

4. Ballard, B. The syntax and semantics of user-defined modifiers in a transportable natural language processor. *Proc. of the 22nd Annual Meeting of the Assoc. Comput. Linguistics*, Stanford University, July, 1984.

5. Valtorta, M. Knowledge refinement in rul bases for expert systems: an application-driven approach. *First Intern. Workshop on Expert Database Systems*, Kiawah Island, S.C., October, 1984.

6. Valtorta, M., B. Smith and D. W. Loveland. The Graduate Course Adviser: a multi-phase rule-based expert system. *IEEE Workshop on Principles of Knowledge-Based Systems*. Denver, December, 1984.

Note: Bruce Ballard, while not funded by this grant this year, was funded last year, and in previous years on an earlier AFOSR grant.