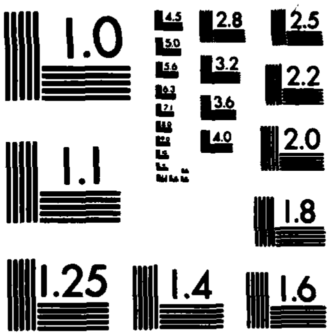
AD-A127 367   A CONCEPTUAL FRAMEWORK AND A HEURISTIC PROGRAM FOR THE   1/1
CREDIT ASSIGNMENT..(U) ARIZONA STATE UNIV TEMPE GROUP
FOR COMPUTER STUDIES OF STRATE.. N V FINDLER ET AL.
UNCLASSIFIED   FEB 83 GCSS-TR-13 AFOSR-TR-83-0289   F/G 12/1   NL

END
FILMED


1.0
4.5
5.0
5.6
6.3
7.1
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

ASU Department of Computer Science

A CONCEPTUAL FRAMEWORK AND A HEURISTIC PROGRAM FOR THE CREDIT ASSIGNMENT PROBLEM

Nicholas V. Findler and Bede McCall
Group for Computer Studies of Strategies
Computer Science Department
Arizona State University
Tempe, AZ 85287

AFOSR-82-0340

TEMPE, ARIZONA 85287

A CONCEPTUAL FRAMEWORK AND A HEURISTIC PROGRAM FOR THE CREDIT ASSIGNMENT PROBLEM

Nicholas V. Findler and Bede McCall
Group for Computer Studies of Strategies
Computer Science Department
Arizona State University
Tempe, AZ 85287

AFOSR-82-0340

Group for Computer Studies of Strategies
Technical Report Number: 13

Departmental Technical Report Number: TR-83-002

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER AFOSR-TR- 83-0289	2. GOVT ACCESSION NO. AD-A127367	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A CONCEPTUAL FRAMEWORK AND A HEURISTIC PROGRAM FOR THE CREDIT ASSIGNMENT PROBLEM		5. TYPE OF REPORT & PERIOD COVERED INTERIM*, 1 JUL 82-30 JUN 83 6. PERFORMING ORG. REPORT NUMBER DEPT TR-83-002; GCSS TR-13
7. AUTHOR(s) Nicholas V. Findler and Bede B. McCall		8. CONTRACT OR GRANT NUMBER(s) AFOSR-82-0340
9. PERFORMING ORGANIZATION NAME AND ADDRESS Group for Computer Studies of Strategies Computer Science Department, Arizona State University, Tempe AZ 85287		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS PE61102F; 2304/A2
11. CONTROLLING OFFICE NAME AND ADDRESS Mathematical & Information Sciences Directorate Air Force Office of Scientific Research Bolling AFB DC 20332		12. REPORT DATE FEB 83 13. NUMBER OF PAGES 18
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

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)

Credit assignment; the Quasi-Optimizer system; automatic computer model generation; multi-level learning programs; Meta-strategies.

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

The authors interpret the task of credit assignment as the ability of a system

(1) to identify and distinguish strategy components;

(2) to associate with such components, in different regions of the domain of confrontation, good and poor conditions of a sequence of actions prescribed by the strategy;

(CONTINUED)

ITEM #20, CONTINUED:

(3) to improve the boundaries of these regions in minimizing the errors of misclassification of strategy actions;

(4) to estimate the overall quality of a strategy being the sum of the qualities of strategy components weighted by the probabilities of employing them;

(5) to provide information for a meta-strategy that shifts the domain of confrontation to those regions in which the strategy studied is most proficient and, thereby, raises the effective quality of the strategy.

The paper describes the program QO-4 --- the fourth module of a large system, the Quasi-Optimizer --- which can accomplish the above tasks within certain limitations.

A CONCEPTUAL FRAMEWORK AND A HEURISTIC PROGRAM FOR THE CREDIT ASSIGNMENT PROBLEM

Nicholas V. Findler and Bede B. McCall
Group for Computer Studies of Strategies
Computer Science Department
Arizona State University
Tempe, AZ 85287

ABSTRACT

The author(s) ~~We~~ interpret the task of credit assignment as the ability of a system:

(i) to identify and distinguish strategy components;

(ii) to associate with such components, in different regions of the domain of confrontation, good and poor outcomes of a sequence of actions prescribed by the strategy;

(iii) to improve the boundaries of these regions in minimizing the errors of misclassification of strategy actions;

(iv) to estimate the overall quality of a strategy being the sum of the qualities of strategy components weighted by the probabilities of employing them; and

(v) to provide information for a meta-strategy that shifts the domain of confrontation to those regions in which the strategy studied is most proficient and, thereby, raises the effective quality of the strategy.

The paper describes the program QO-4 -- the fourth module of a large system, the Quasi-Optimizer -- which can accomplish the above tasks within certain limitations.

1. INTRODUCTION

Humans and machines make a *vast number of decisions* before an action is selected. In turn, only a *long sequence of actions* may at times lead to some tangible result that proves to be of some indisputable quality, ranging from excellent to disastrous. For example, sophisticated chess players go through probably ***hundreds of decision junctions*** for each move. A game ends with a win, draw or loss after ***dozens of moves.*** Which decisions and to what degree are responsible for the final outcome? (Note that even a brilliant play, so recognized by expert annotators, may end with a loss.)

This issue, usually referred to as the "credit-assignment problem," has drawn much attention from the early times of AI on. (See, e.g., [1].) A recent article [2} discusses a learning technique to identify useful conditions for applying operators in a heuristic search for solutions. Positive credit is assigned to solution paths and negative credit to failures; the *extent* of the credit depending on the level of success or otherwise. The approach described looks promising and needs to be implemented for a non-trivial task environment for meaningful evaluation.

We interpret the task of credit assignment as the ability of a system

(i) to identify and distinguish strategy components;

(ii) to associate with such components, in different regions of the domain of confrontation, good and poor outcomes of a sequence of actions prescribed by the strategy;

(iii) to improve the boundaries of these regions in minimizing the errors of misclassification of strategy actions;

(iv) to estimate the overall quality of a strategy; and

(v) to provide information for a meta-strategy which shifts the domain of confrontation to those regions in which the strategy studied is most proficient and, thereby, raises the effective quality of the strategy.

2. ON STRATEGIES, THEIR COMPUTER REPRESENTATION AND THE QUASI-OPTIMIZER SYSTEM

A *strategy* is a decision-making mechanism that observes and evaluates its environment, and prescribes in response to it an *action*. This action, at the simplest level, does not change for the same environment over time, is a single and one-step response.

We have extended this concept in several directions. *Learning strategies* no longer are static. They improve the technique of evaluating the environment as well as the selection of the action, on the basis of experience. The single (that is, one-dimensional) action can be replace by a *set of* (that is, multi-dimensional) *actions*. Instead of a one-step (momentary) action, we may have a *sequence of actions* that are unordered, weakly or strongly ordered over time. Finally, the decision variables defining the environment may also include descriptors that characterize relevant aspects of the *history of the environment*.

All these extensions make our studies more realistic, taking into account learning strategies, which can issue also multi-dimensional responses to complex environments. The actions may be the results of long-range planning processes and are based on both short-term and long-term considerations (tactical and strategic objectives, respectively).

Figure 1 about here.

We represent static strategies prescribing simple actions through decision trees (DT's)--see Figure 1. We note here only one representational extension. We have developed a program that "freezes" the learning component of such a strategy and

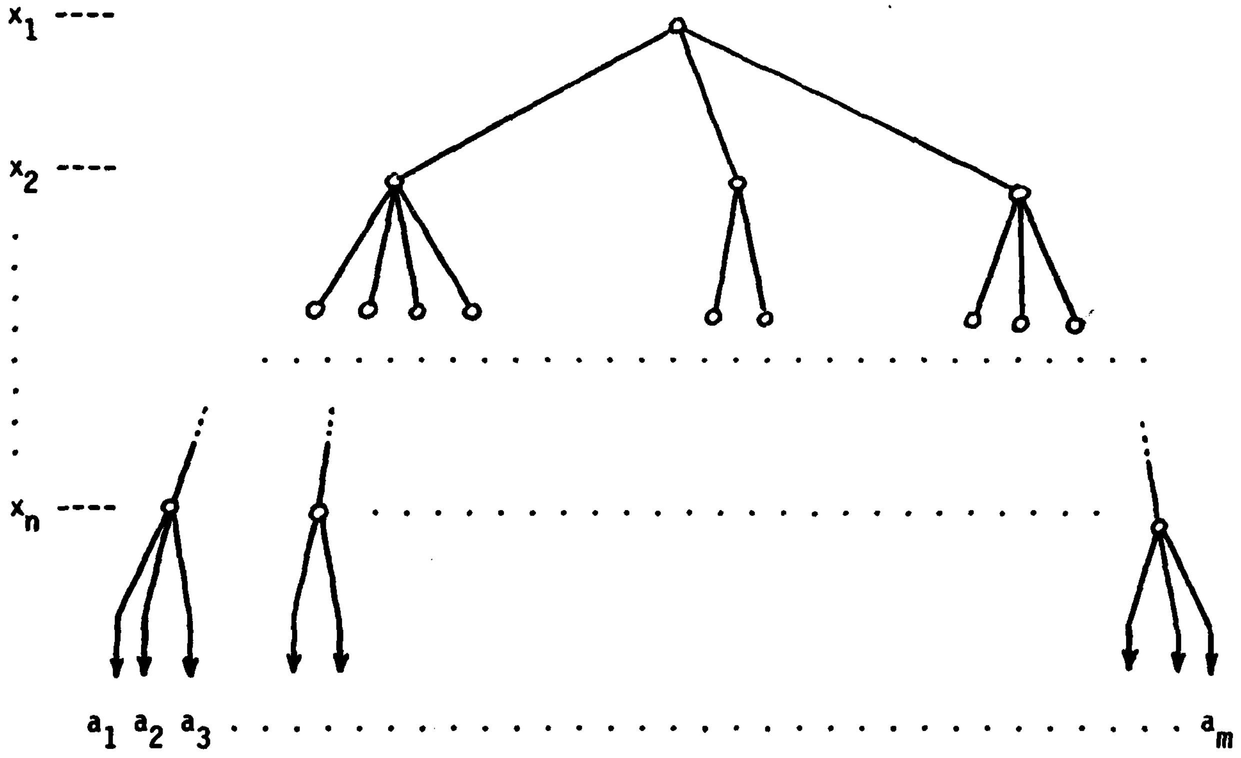


FIGURE 1

takes a "snapshot" of it in the form of a DT [3]. Another module [4] receives such a sequence of snapshots and, if it is statistically justified, computes the asymptotic form to which the sequence converges. We also note that the automatic generation of the computer model, the snapshot, can be done by the system being a passive observer or "under laboratory conditions," according to some experimental design. The experiments in the latter case are specified in one of three different ways:

(i) in an exhaustive manner when every level of a decision variable is combined with every level of the other decision variables;

(ii) by a binary chopping technique while relying on the assumption of a monotonically changing response surface;

(iii) according to a dynamically evolving design in which the levels selected for the decision variables, and the length of the whole experimentation, depend on the experimental results obtained up to that point [5]. This module minimizes the total number of experiments for a given level of precision.

The programs outlined above and the one discussed in this paper are some of the modules of a large-scale system called the *Quasi-Optimizer* (QO). The QO has three major objectives [6]:

(a) to observe and measure adversaries' behavior in a competitive environment, to infer their strategies and to construct a computer model, a ***descriptive theory***, of each;

(b) to identify strategy components, evaluate their effectiveness and to select the most satisfactory ones from a set of descriptive theories;

(c) to combine these components in a quasi-optimum strategy that represents a ***normative theory*** in the statistical sense.

The present program is the fourth module of QO and we shall refer to it as QO-4.

3. SOME ASSUMPTIONS AND TERMINOLOGY

We assume the existence of a *critic* that returns a single scalar value, the *quality measure* of the short- and long-term consequences of a certain strategy action responding to a given environment.

A *situation* is a state of the environment described by certain current and past values of a strategy-dependent set of decision variables. A *strategy response type* is a set of responses, each member of which is in close proximity to the others. A *strategy component* is a collection of situations grouped together on the basis of a common response type.

There are two kinds of *environmental features* and both refer to a single strategy component. *Characteristic features* are descriptive of a strategy component, without reference to other situations and response types. *Contrasting features* distinguish between two strategy components sharing the same strategy response type.

One can view the concept of a feature in two different ways. First, a *generalized view* holds that a feature should comprise the broadest set of factors that accurately describes a strategy component. Second, according to a *specialized view*, features are made of factors that have an absolute certainty of being true for all situations in a strategy component. Which view should be adhered to is a question of the application as well as a trade-off between precision and computational complexity. After some experimentation, we have found a reasonable compromise between the two views that is both effective and efficient. A feature is the Boolean AND of the decision variable subranges most frequently overlapped within a strategy component.

Contrasting features can also be approached from two angles, regardless of which view we assume concerning characteristic features. Contrast can be stated in either "positive" or "negative" terms although both can lead to the same computational results. With positive terms, we collect characteristic features which are held exclusively by each feature set. The negative terms idea describes the intersection of the two feature sets. Depending on whether the comparison between a situation and positive or negative terms is cheaper on the average, we --and QO-4-- can choose one or the other.

Finally, some notational conventions. The components of a situation vector, $\underline{s}_i$ are the values of the decision variables characterizing the situation: $\underline{x}_1^{(i)}, \underline{x}_2^{(i)}, \underline{x}_n^{(i)}$. The situation corresponds to a pathway on the DT, $\underline{r}_i$. It leads to an action at the leaf level, $\underline{a}_j$. The $\underline{k}$ -th response type to which, say, $\underline{a}_j$ belongs is $\underline{A}_k$ The set of pathways leading to an action within $\underline{A}_k$ is denoted by $\underline{R}_k$. The quality of a given action, $\underline{a}_j$, in a given situation is $q(r_i,a_j)$ --- q_{ij} for short. This measure is, in the first approximation, independent of the strategy. (Long-range planning by the strategy and, correspondingly, long-term consequences of the strategy response can affect the quality measure.)

We shall be concerned with three subranges of the quality measure: 'bad', 'neutral' and 'good'. If the total, normalized quality range is (0,100), the respective subranges are: (0,B), (B,G) and (G,100) -- with boundary points $\underline{B}$ and $\underline{G}$ to be determined by a learning process described later. The corresponding subsets of the pathway set $\underline{R}_k$ are $\underline{R}_k^{(B)}$, $\underline{R}_k^{(N)}$ and $\underline{R}_k^{(G)}$. When reference is made to one of these subsets, without regard to quality values, we shall use the notation $\underline{R}_k^{()}$.

4. THE ALGORITHM AND A WALK-THROUGH EXAMPLE

In explaining the algorithm, we shall make reference to a sample DT shown in Figure 2. Note that it is an extremely simple tree; the corresponding strategy has only two decision variables and twelve possible actions.

Figure 2 about here.

QO-4 begins by extracting the characteristic features of the individual (ungrouped) situations. This operation consists of simply forming the Boolean AND of the subranges along the individual pathways. The results are shown in Table 1.

Table 1 about here.

The fifth column in Table 1 contains the estimates of the probability of given situations occurring. Unless the module QO-1 operates as a passive observer in generating a computer model (snapshot) of the strategy to tell better, we assume a uniform probability distribution of the decision variables over their total range, normalized to (0,128). The probability of a situation occurring is then the product of the relative length of the subranges through which the pathway in question passes. (We note that QO-4 can also use observed probability values if provided by QO-1, or assume that each pathway is equiprobable.) These probability figures then appear in the computation of the estimated quality of the strategy.

$$Q^{*}=\sum_{(i)} p(\hat{s}_i)\bullet q(r_i,a_j)$$

Next, the pathways are assembled into strategy components. Using ± 2 as a "tolerance level" for a response type, we get four strategy components. These and their characteristic features are shown in Table 2.

Table 2 about here.

The Q^{*} value calculated above will be the initial boundary point between the "bad" and "good" quality subranges, and $\mathbb{R}_k^{(N)}$ will at first be an empty subset. QO-4 then calculates the characteristic features of $\mathbb{R}_k^{()}$ and the quality for each strategy component. A learning process has been implemented to minimize the

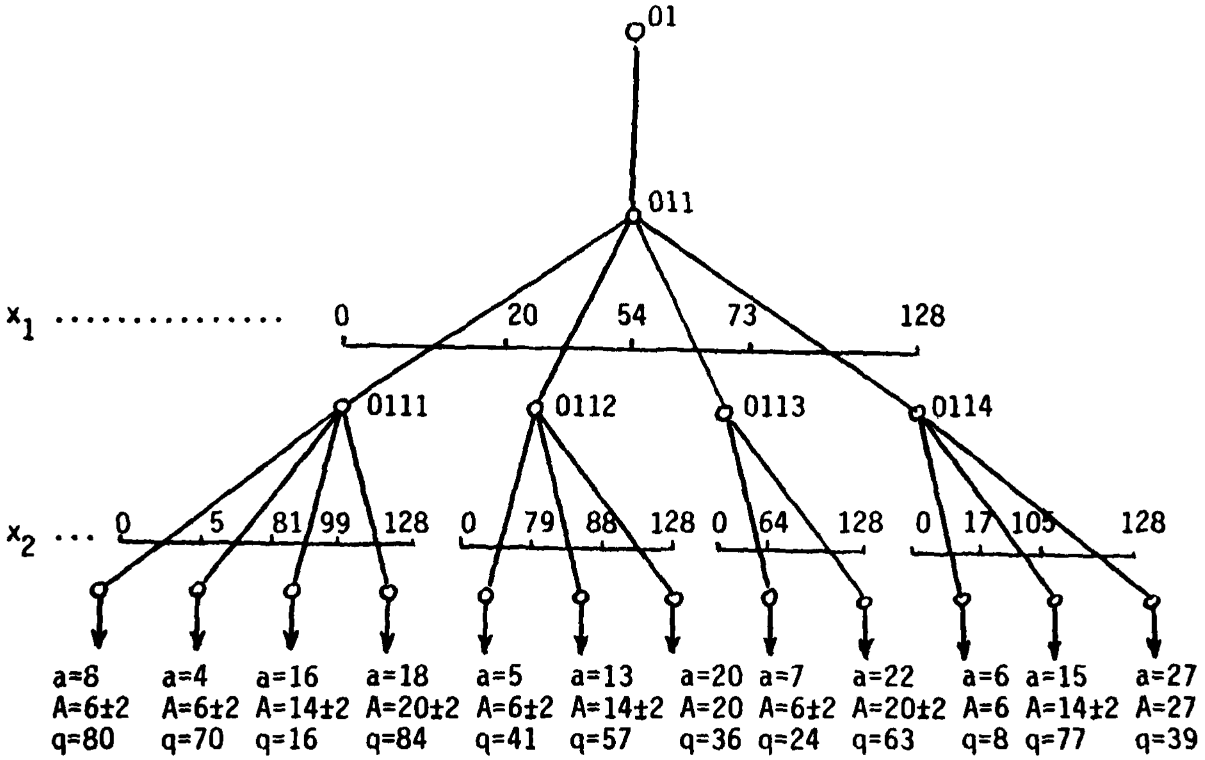

01
011
x_1 ....
0
20
54
73
128
0111
0112
0113
0114
x_2 ...
0
5
81
99
128
0
79
88
128
0
64
128
0
17
105
128
a=8 A=6±2 q=80
a=4 A=6±2 q=70
a=16 A=14±2 q=16
a=18 A=20±2 q=84
a=5 A=6±2 q=41
a=13 A=14±2 q=57
a=20 A=20 q=36
a=7 A=6±2 q=24
a=22 A=20±2 q=63
a=6 A=6 q=8
a=15 A=14±2 q=77
a=27 A=27 q=39

FIGURE 2

situation numbers	subranges for decision variable		strategy response	"estimates" of prob. of a situation occurring	quality of action
i	x_1	x_2	a_j	$p^*(s_i)$	$q(r_i,a_j)$
1	(0 19)	(0 4)	8	0.006	80
2	(0 19)	(5 80)	4	0.091	70
3	(0 19)	(81 98)	16	0.021	16
4	(0 19)	(99 128)	18	0.036	84
5	(20 53)	(0 78)	5	0.161	41
6	(20 53)	(79 87)	13	0.018	57
7	(20 53)	(88 128)	20	0.084	36
8	(54 72)	(0 63)	7	0.073	24
9	(54 72)	(64 128)	22	0.074	63
10	(73 128)	(0 16)	6	0.057	8
11	(73 128)	(17 104)	15	0.330	77
12	(73 128)	(105 128)	27	0.081	39

$Q^*=56.3$

TABLE 1

misclassification of the quality of a pathway (a situation) on the basis of its features. The boundary points $\underline{B}$ and $\underline{G}$ no longer coincide. The "neutral" quality subrange between them is eventually filled with situations that would have changed from "bad" to "good" or the other way around, in the course of learning iterations. In analogy with the two types of errors in statistical hypothesis testing, the optimum location of $\underline{B}$ and $\underline{G}$ will minimize Type II error (accepting the quality of a wrong pathway) while Type I error is eliminated (rejecting the quality of a right pathway), at the cost of pushing the "dubious" pathways into the "neutral" quality.

The final results (omitting average qualities, probabilities of occurrence and R_k^{ℓ} values of strategy components) are shown in Table 3.

Table 3 about here.

5. THE PROGRAM QO-4 AND STATISTICAL PATTERN RECOGNITION

We have used quality as a linear discriminant function (see, e.g. [7]) for classifying strategy situations. To overcome measurement errors and other types of noise, and to reduce computational complexity, we have introduced the concept of strategy components. Strategy situations are classified on the basis of computational results involving strategy components. The price we have paid for the transition between the two is some "fuzziness" in the discrimination criteria but the reliability of the credit assignment has gained considerably.

We have to characterize the cases in which our approach is likely to be effective. It seems plausible that whenever there is a significant correlation between the environmental conditions and the quality function, QO-4 will produce useful results. This implies that the strategy under study operates "reasonably" and in an environment which has a large element of rule-bound behavior, although the exact nature of the rules need not be known. It is under such conditions that the quality

Quality-clustered strategy components	Characteristic features of subranges for		
	x_1	AND	x_2
$R^{(B)}$	$[(0\ 19) \vee (54\ 128)]$	$\wedge$	$[(0\ 63) \vee (81\ 98)]$
$R^{(N)}$	$[(20\ 53) \vee (73\ 128)]$	$\wedge$	$[(0\ 78) \vee (88\ 128)]$
$R^{(C)}$	$(0\ 128)$	$\wedge$	$(0\ 128)$
Contrasting features (discriminating in this case only $R^{(B)}$)	$[(0\ 19) \vee (54\ 128)]$	$\wedge$	$[(0\ 63) \vee (81\ 98)]$

TABLE 3

values are, to a large extent, causally related to what and why the strategy does.

6. TWO POSSIBLE EXTENSIONS OF WORK

We select two possible extensions of work out of many. First, the quality function could return a vector value instead of a scalar. Each component would describe how good a particular action is in regard to one single subgoal. (Think of a chess move's contribution to King's safety, center control, mobility of pieces, etc.) Such a case represents no conceptual problem for QO-4. It would go through the analysis of credit assignment for each of the vector components.

The second extension is based on an assumption outlined before. Namely, all strategies are "reasonable" and are, therefore, somewhat similar. If QO-4 processes a *set* of such strategies and forms more reliable probability estimates, it can follow the logic of a Bayesian classifier with very powerful learning features. Any discussion of this, however, is beyond the scope of this paper.

7. SUMMARY

QO-4, a program aimed at the credit assignment problem is limited in several ways. The nature of the strategies it can deal with, the environment, and their simplified and potentially error-prone representation reduce the scope of the system as well as introduce inherent imprecisions. Within these limitations and for the tasks for which the QO system has been designed, QO-4 accomplishes its goal.

The generality of the approach has necessitated certain "safe" choices--such as the way in which the discriminant function interacts with the system and the averaging of its value over strategy components--in opposition to more informative but domain-dependent possibilities.

8. ACKNOWLEDGMENTS

Neal Mazur and Bob Cromp have converted the system from ALISP to MACLISP. Tim Bickmore did the word-processing for the article. We all are grateful for the support from AFOSR Grants 81-0220 and 82-0340.

9. REFERENCES

[1] Minsky, M.: Steps Towards Artificial Intelligence (In Feigenbaum and Feldman (Eds.): *Computers and Thought*. McGraw-Hill: New York, 1963).

[2] Sleeman, D., P. Langley and T.M. Mitchell: Learning from solution paths: An approach to the credit assignment problem *(The AI Magazine, 3,* pp. 48-52, 1982).

[3] Findler, N.V. and J.P. Martins: On automating computer model construction--The second step toward a Quasi-Optimizer system *(J. of Info. and Optimization Sciences, 2,* pp. 119-136, 1981).

[4] Findler, N.V., N. Mazur and B. McCall: A note on computing the asymptotic form of a limited sequence of decision trees (To appear in *Information Sciences)*.

[5] Findler, N.V. and R.F. Cromp: An artificial intelligence technique to generate self-optimizing experimental designs (Submitted for publication).

[6] Findler, N.V. and J. van Leeuwen: The complexity of decision trees, the Quasi-Optimizer and the power of heuristic rules *(Info. and Control, 40,* pp. 1-19, 1979).

[7] Devivjer, P.A. and J. Kittler: *Pattern Recognition: A Statistical Approach* (Prentice-Hall: Englewood Cliffs, NJ, 1982).

LEGEND FOR FIGURES AND TABLES:

FIGURE 1 - A decision tree with n decision variables, $x_1, \ldots, x_n$, and m responses, $a_1, \ldots, a_m$.

FIGURE 2 - An exemplary decision tree used in the explanation of the algorithm. Each node is marked with a "Dewey decimal" index. The subranges of each variable's normalized range, (0, 128), are shown. There are two decision variables (x_1 and x_2), twelve different responses (a's), four response types (A's), and twelve quality measures (q's) normalized to (0, 100).

TABLE 1 - Various values associated with all possible situations are shown on the decision tree of Figure 2. The estimated quality of the strategy is $Q^* = \sum_{(i)} p^*(s_i) * q(r_i, a_j)$.

TABLE 2 - Strategy components and their characteristic features.

TABLE 3 - Some of the final results of QO-4 working on the exemplary decision

FILM