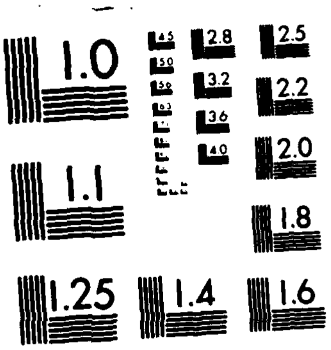
AD-A127 764   AN ARTIFICIAL INTELLIGENCE TECHNIQUE TO GENERATE   1/1
SELF-OPTIMIZING EXPERIME..(U) ARIZONA STATE UNIV TEMPE
GROUP FOR COMPUTER STUDIES OF STRATE..
UNCLASSIFIED   N V FINDLER ET AL. 01 FEB 83 GCSS-TR-12   F/G 6/4   NL


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

AN ARTIFICIAL INTELLIGENCE TECHNIQUE TO GENERATE SELF-OPTIMIZING EXPERIMENTAL DESIGNS

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

Group for Computer Studies of Strategies
Technical Report Number: 12

Departmental Technical Report Number: TR-83-001

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR- 83-0337	2. GOVT ACCESSION NO. AD A127 764
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) AN ARTIFICIAL INTELLIGENCE TECHNIQUE TO GENERATE SELF-OPTIMIZING EXPERIMENTAL DESIGNS	5. TYPE OF REPORT & PERIOD COVERED INTERIM, 1 JUL 82-30 JUN 83
	6. PERFORMING ORG. REPORT NUMBER DEPT TR-83-001; GCSS TR-12
7. AUTHOR(s) Nicholas V. Findler and Robert F. Cromp	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 1 FEB 83
	13. NUMBER OF PAGES 10
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)

Quasi-optimizer; automatic computer model generation; statistical design of experiments; automatic experimental design generation; generalized sequential testing.

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

The paper describes a completed and independent module of a large-scale system, the Quasi-Optimizer (QO). The QO system has three major objectives: (1) 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; (2) to identify strategy components, evaluate their effectiveness and to select the most satisfactory ones from a set of computed descriptive theories; and (3) to combine these components in a quasi-optimum strategy that represents a normative theory in the statistical sense. (CONTINUED)

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

83 05 06-115

ITEM #20, CONTINUED:

The measurements on the input strategies can take place either in a sequence of confrontations unperturbed by the QO or, for efficiency's sake, in a series of environments specified according to some experimental design. The module completed first, QO-1, can perform the experiments either in an exhaustive manner - when every level of a decision variable is combined with every level of the other decision variables - or, in relying on the assumption of a monotonically changing response surface, it uses the binary chopping technique.

The module discussed here, QO-3, does not assume monotonic response surfaces and can deal also with multidimensional responses. It starts with a (loosely) balanced incomplete block design for the experiments and computes dynamically the specifications for each subsequent experiment. Accordingly, the levels of the decision variables in any single experiment and the length of the whole sequence of experiments depend on the responses obtained in previous experiments. In general, QO-3 is an on-line, dynamic generator of experimental design that minimizes the total number of experiments performed for a predetermined level of precision.

AN ARTIFICIAL INTELLIGENCE TECHNIQUE TO GENERATE SELF-OPTIMIZING EXPERIMENTAL DESIGNS

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

ABSTRACT

The paper describes a completed and independent module of a large-scale system, the Quasi-Optimizer (QQ). The QQ system has three major objectives: (i) 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; (ii) to identify strategy components, evaluate their effectiveness and to select the most satisfactory ones from a set of computed descriptive theories; (iii) to combine these components in a quasi-optimum strategy that represents a normative theory in the statistical sense.

The measurements on the input strategies can take place either in a sequence of confrontations unperturbed by the QQ or, for efficiency's sake, in a series of environments specified according to some experimental design. The module completed first, QQ-1, can perform the experiments either in an exhaustive manner -- when every level of a decision variable is combined with every level of the other decision variables -- or, in relying on the assumption of a monotonically changing response surface, it uses the binary chopping technique.

The module discussed here, QQ-3, does not assume monotonic response surfaces and can deal also with multidimensional responses. It starts with a (loosely) balanced incomplete block design for the experiments and computes dynamically the specifications for each subsequent experiment. Accordingly, the levels of the decision variables in any single experiment and the length of the whole sequence of experiments depend on the responses obtained in previous experiments. In general, QQ-3 is an on-line, dynamic generator of experimental design that minimizes the total number of experiments performed for a predetermined level of precision.

1. INTRODUCTION

The Quasi-Optimizer (QQ) system [1, 2] observes and measures adversaries' behavior in a competitive environment, infers their strategies, and constructs a computer model (a "descriptive theory") of each. By evaluating the effectiveness of the components of these strategies and selecting the most satisfying ones (credit assignment), it generates a "normative theory" which is an optimum strategy in the statistical sense. The measurement of the adversaries' behavior can take place either in a sequence of unperturbed confrontations or under "laboratory conditions" when the environment for each confrontation is specified according to some experimental design. We shall be concerned with the second mode of operation in this paper.

The first of six fairly independent modules of the QQ system, QQ-1, constructs a descriptive theory of static strategies given as black-box programs impenetrable by QQ-1 [3]. It also identifies which of all possible decision variables are relevant for the strategy being modelled. The program can use either an exhaustive search pattern or a binary chopping technique in the space of decision variables while carrying out a sequence of controlled experiments on the strategy. As an inductive discovery feature, it can also correlate certain stochastic consequences of the strategy with subranges of values of each decision variable. The strategy response surface is assumed by QQ-1 to be weakly monotonic.

The present paper deals with a significant generalization of the QQ-1 program. The module QQ-3 is designed to minimize the total number of experiments while maintaining a user-specified minimum level of precision in identifying the strategy response, R, over the whole space of decision variables. The response surface need not be monotonic now. The design of each experiment, after an exploratory phase, depends on the results of the experiments obtained up to that point.

The program QQ-3 is completely general purpose. However, because of the specific context of the QQ project, we shall use the terms 'decision variables' for the control variables in experiments, and 'strategy response' for the scalar or vector entity that is the outcome of the experiments.

2. ON STRATEGIES AND THEIR COMPUTER REPRESENTATION

A strategy is considered at its simplest level to be a decision making mechanism that observes and evaluates its environment, and prescribes in response to it a single, one-step action. We can extend this concept in various directions. The single (that is, one-dimensional) action can be replaced by a *set of* (that is, multidimensional) *actions*. The one-step (momentary) action can be replaced by a *sequence of actions*, unordered, weakly or strongly ordered, over time. Furthermore, the decision variables defining the environment may also include descriptors characterizing relevant aspects of the *history of the environment*. These ideas make our studies more realistic in taking into account multidimensional strategy responses to complex environments, long-range planning, tactical and strategic considerations (with reference to short-term and long-term objectives, respectively). We can study automatically generated "methods", in which goals and current features of the environment are associated with sequences of actions, and "blueprints", in which goals are presented as desired features of the environment. We also distinguish between *static* and *dynamic strategies*. The latter are either controlled by a *learning mechanism* (to improve performance or to adapt to new environments), or exhibit periodic or random *fluctuations*. (See [4] for a detailed discussion.)

We have selected the *decision tree* (DT) as an efficient and effective representation of simple, single-action strategies [1]. (See Fig. 1.) We have also shown that DTs are equivalent in power to production systems but can be modified more easily and their scope of representational validity can be extended as needed. These extensions are as follows:

FIGURE 1 ABOUT HERE

.When the strategy response is a vector quantity, each of its components requires a separate DT. (We are currently studying techniques to eliminate any redundancy inherent in cases in which the vector components are correlated.)

.A time-sequence of actions can be attached to the leaf-level, instead of one-step strategy responses, to describe the result of strategic planning.

.Judiciously chosen decision variables can characterize the relevant aspects of the history of a confrontation or of the development of an environment.

.A learning strategy is represented by a sequence of DTs, each being a "snapshot" taken of the strategy, with the learning component frozen, at different time points. We have devised an algorithm, the *QQ-2* module [4], that computes the asymptotic form of the sequence of DTs, when the result is statistically valid. This extrapolated DT is then used as one of the input strategies in the computation of the normative strategy.

3. *THE QQ-3 PROGRAM*

We can explain the *QQ-3* best by going through its phases of operation in a chronological order.

3.1. *The User Input*

The whole program is highly interactive and relies on the user's advice when feasible. As described below, the first, exploratory phase of the program specifies decision variable levels according to a loosely balanced incomplete block design, a term to be explained later. The user first has to input a so-called *reduction factor*, *f*, which is the ratio between the number of exploratory experiments and the number of all possible experiments. The latter is the product of the number of meaningfully distinct levels of every decision variable -- essentially the *cardinality* of the experiment. The reduction factor provides the user with control over the usual trade-off in experimentation between cost and precision.

The user is then asked to specify ΔR, the precision desired (or error tolerated). This is then considered by the system as the minimum discernible difference between the strategy responses given at two adjacent experimental points in the decision variable space. Therefore, if there is reason to assume a weakly monotonic response surface, the latter is considered "flat" between adjacent points whenever the response values at such points differ by no more than ΔR. (Our program repeats the experiments at the two points once more and also checks the response value in the midpoint because of possible non-monotonicity of the surface and the usually stochastic nature of the environment.)

Next, the user inputs information about each decision variable. This consists of its name, type, range, and the initial estimate of the number of levels it assumes. There are three types of variables:

(i) Numerical in which case the range of values is normalized to (0, 128). The user estimates how sensitive the strategy response is to changes in the variable in question. Higher sensitivity, i.e. more rapid changes, would require more levels in the variable. The user must also specify the maximum meaningful resolution (MMR), which is the smallest discernible difference between the values of the variable. In other words the grid size along that dimension must be at least as large as MMR.

(ii) Ordered categorical variables assume symbolic values which are, by their nature, ordered. Examples are rank numbers, the days of the week, musical notes, even colors when their respective wave lengths have some significance. The user may enter, for example, ((COLOR (RED ORANGE YELLOW GREEN BLUE INDIGO VIOLET)). The system again maps the range of the user-specified values onto (0, 128). He also provides an MMR value to express how "influential" the variable is with regard to changes in the strategy response. Wisely used, the user can control through MMR the number of experiments until more information becomes available about the nature of the response surface. The highest number of levels of a numerical or ordered categorical variable, NL, is related to MMR as

$$MMR = 128 / (NL - 1)$$

(iii) Unordered categorical variables, too, assume symbolic values but these have no meaningful order. The user may, for example, specify ((ANSWER (YES NO)) (SPICES (SALT PEPPER PAPRIKA))). There is no MMR specifiable here. Unordered categorical variables are treated differently; all levels given are used exhaustively, as explained below.

We should point out that QO-3 is robust enough to rectify user errors about the importance of individual decision variables. The program will trim and add levels as the experimentation proceeds and the shape of the response surface emerges. However, time and cost of experiments are saved when the user's estimates are sound.

Finally, the system computes all acceptable *base-unit* values and, if there is more than one, it asks the user to select one. The base-unit is the greatest common divisor of the number of levels of all numerical and ordered categorical variables. (The unordered categorical variables are always exhaustively searched.) An 'acceptable' base-unit is usually a compromise representing the smallest number of levels added to those originally specified by the user, over all affected variables.

3.2. *Block Design for the Exploratory Phase*

The *balanced incomplete block design* (BIBD) is used in controlled experiments to reduce their total number while maintaining the symmetry of effects of two individual and potentially interacting independent variables on one dependent variable (see [5] for details). Unfortunately, it is not possible to construct a BIBD for any number of levels even in the two-dimensional case, and the constraints employed have no obvious counterparts in higher dimensions. These reasons have led us to its generalized concept, the *loosely balanced incomplete block design* (LBIBD).

LBIBD ensures that a statistically reliable sample is selected of all possible combinations of the levels of the decision variables. The size of the sample is the fraction, specified by the user as the reduction factor, of the cardinality of the experiment. The design must satisfy two constraints:

(i) Each level of a variable appears (approximately) equal number of times;

(ii) Each level of a variable is combined with each level of another variable (approximately) equal number of times, for all pairs of variables.

The term "loosely balanced" is due to the fact that another rule concerning the symmetry between multiple co-occurrences of levels, satisfiable only in certain instances of the two-dimensional problem, has been relaxed and used only when possible. The following concepts will be necessary in explaining the other phases of *QO-3*:

Let the reduction factor be given as a fraction of lowest terms, f=a/b; the size of the base-unit be *n*; the number of decision variables of the numerical and ordered categorical types be *d*; and the number of unordered categorical variables be *u*. Let us also define a few terms. A 'chip' consists of (d-1) indices (or level numbers) where an index value falls in the range (1, n) inclusively. A 'basic block' consists of f*n**(d-1) chips. An 'extended block' is the result of 'spawning' the appropriate chips of a basic block. 'Spawning' means repeating the chip along the dimension of a decision variable whose number of levels is a multiple of the base-unit. A 'test vector' is determined by (d+u) indices and represents the specification of one experiment. Finally, the 'initial test basis' consists of a set of test vectors computed by the LBIBD-generator for the exploratory phase of *QO-3*.

Using number-theoretical arguments, it can be shown that if *xj* is an index of the *j*-th variable, then the indices of the (d-1) variables of a basic block must satisfy

$$(x_1 + x_2 + \dots\dots + x_{d-1}) \bmod b < a \qquad (1)$$

Inequality (1) defines stripes perpendicular to the principal diagonal of the block. These stripes can be eliminated and the experiments "randomized" (spread out) with

$$[p(x_1) + p(x_2) + \dots\dots + p(x_{d-1})] \bmod b < a \qquad (2)$$

where p is a permutation operator on the values 0, 1, ... b-1. We have chosen to use the multiplying factor (d-i) for x_i as the respective permutation operator.

3.3. Sensitization as the Exploratory Phase

The process of sensitization is the exploratory phase of QO-3. Its task is to find out where the initial test basis has to be refined along the dimension of every decision variable. (Note that, under ideal conditions, the final grid is such that the difference in response values at adjacent points is identically equal to ΔR over the whole domain of decision variable space.) We describe the process and the underlying heuristic through an example. Suppose one of the d variables, v_i, was specified by the user to have five levels initially. Its normalized values are (0 32 64 96 128). Also, assume it has an MMR of 8. The system first considers the levels 0 and 32. The extended block specifies sets of values of the other (d-1) decision variables at which experiments are performed while the value of v_i is held at 0 and 32, respectively. Accordingly, two groups of response values are obtained, one for v_i=0 and the other for v_i=32. The program forms the average of each group of values. If the difference between the two averages is less than ΔR, the subrange (0, 32) is "monotonically sensitized", i.e. there is no need to refine it if the response surface is assumed to be weakly monotonic. If this assumption is not held, the midpoint v_i=16 is selected. The corresponding average response value is then compared with those for the two endpoints of the subrange. If the respective differences are both less than ΔR, the subrange is "completely sensitized". Otherwise, the midpoint is added to the values of v_i as a level to be used in the completing phase of experimentation. The subranges are halved further whenever the results warrant it -- as long as the length of the subrange is no less than MMR, in our example 8. The same procedure is followed by all subranges of v_i, and then for each of the other decision variables.

Finally, we note that the response values are naturally kept after their averages are formed -- they are needed also in the completing phase of experimentation.

3.4. The Completing Phase of Experimentation

When all decision variables have been sensitized, the experiments specified by all computed test vectors are performed. (There is no saving possible for the unordered categorical variables. The whole process has to be repeated for each value of every such variable.)

Finally, QO-3 builds a DT of the results of computations. The paths go through the subrange of the variables, and the response value is attacled to the leaf level.

4. SOME RESULTS

As a final test run, we have defined a response surface as a function of 2 numerical variables, 1 ordered and 1 unordered categorical variables with the following conditions:

```
IF (state = solid)
   THEN IF $t \cdot d < 3000$ THEN response = $\sqrt{t \cdot d}$
                         ELSE response = $(d/30)\sqrt{t \cdot d}$.
IF (state = liquid)
   THEN IF $t < 100$ THEN response = $t \cdot d$
                      ELSE response = $(t/100) \cdot t \cdot d$.
IF (state = gas)
   THEN IF (gas is radio-active) THEN response = $5 \cdot t \cdot d$
                                 ELSE response = $(5/2) \cdot t \cdot d$.
```

The user has specified the following values: $f = 2/5$, $\Delta R = 2000$,

Variable Name	Type	Range	MMR	Number of Init. Levels
temperature (=t)	numer.	(0..200)	10	6
duration (=d)	numer.	(0..60)	2	4
state	ord. cat.	(solid..gas)	"64"	3
radio-active	unord. cat.	(yes,no)	--	2

QO-3, written in MACLISP, took 31 seconds on a Honeywell Level 68/80 processor to design a total of 434 experiments (out of 3906 possible ones). The actual maximum difference in response values at adjacent points was $\Delta R_M = 1728$.

5. ACKNOWLEDGEMENTS

Paul Duerig, Bill Chang, and Richard Wozniak have also contributed to the design and programming of QO-3 at an earlier stage. George Sicherman has helped us in deriving the method for LBIBD. Ron Lo has done the word processing work for this article. We are all indebted for the support by AFOSR Grants 81-0220 and 82-0340.

6. REFERENCES

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

[2] Findler, N.V.: An overview of the Quasi-Optimizer system (Submitted for publication).

[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 Opt. 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 (Submitted for publication).

[5] John, P.W.M.: Incomplete Block Designs (Marcel Dekker: New York, 1980).

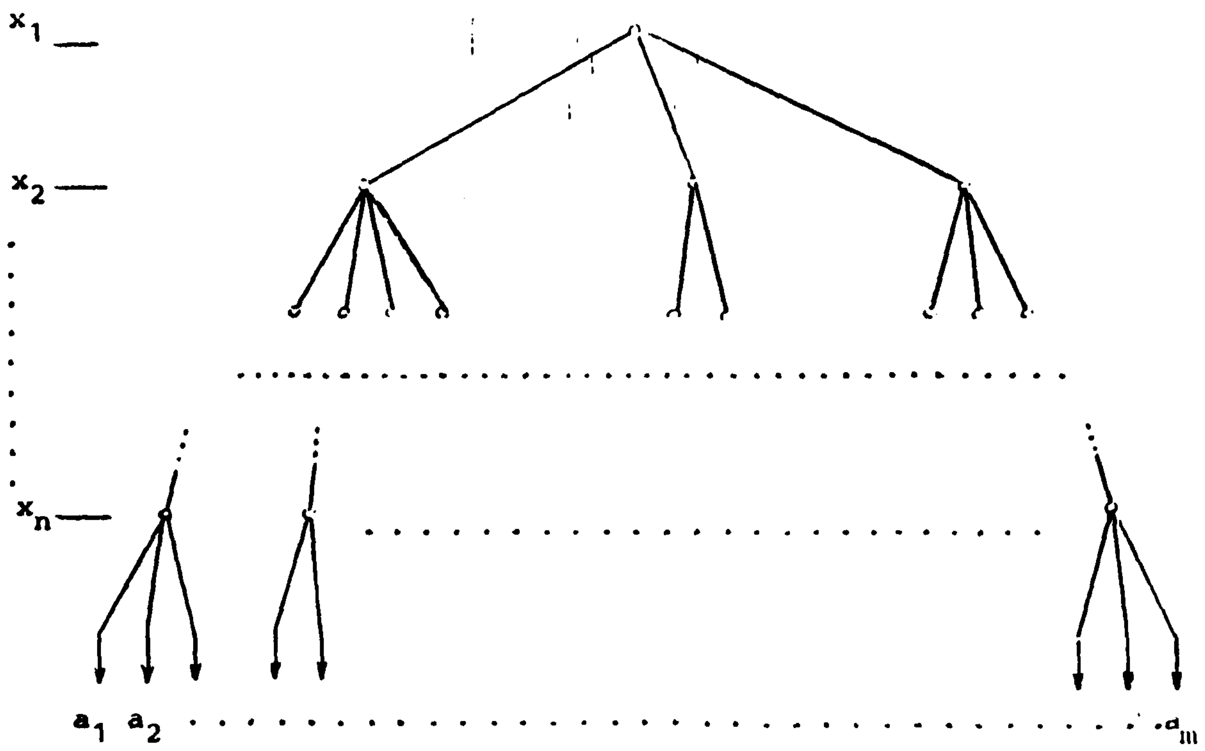


Figure 1

A decision tree with $\underline{n}$ decision variables,

$\underline{x_1}, \ldots \underline{x_n}$, and $\underline{m}$ responses (actions), $\underline{a_1}, \ldots \underline{a_m}$

LME

– 8