UILU-ENG-86-1755

Reports

of the

Intelligent
Systems
Group

A RULE EXERCISER FOR KNOWLEDGE BASE ENHANCEMENT IN EXPERT SYSTEMS

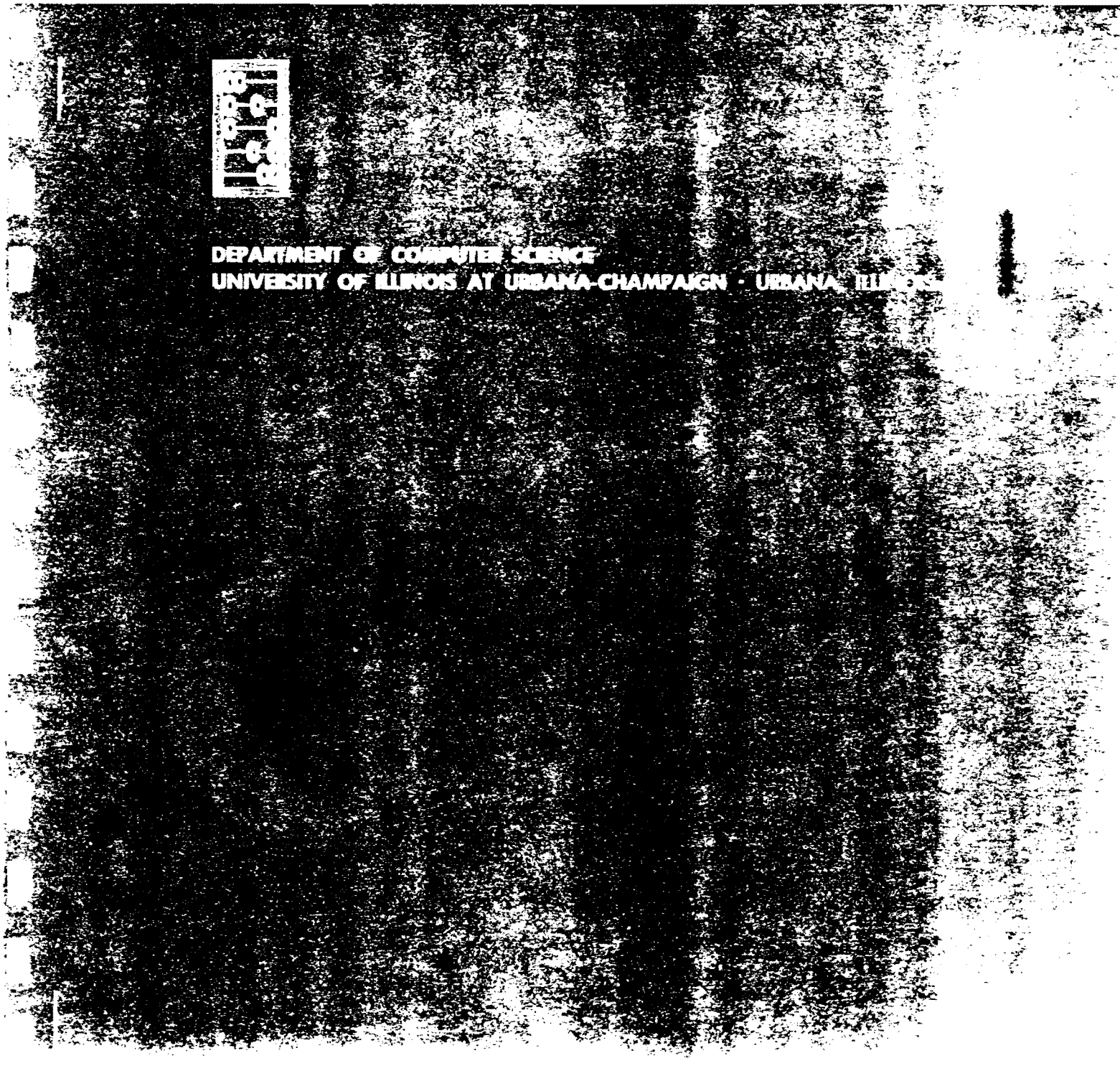
by

Carl Thomas Uhrik

September 1986

File No. UIUCDCS-F-86-969

ISG 86-23



File No. UIUCDCS-F-86-969

A RULE EXERCISER FOR KNOWLEDGE BASE ENHANCEMENT IN EXPERT SYSTEMS

BY

CARL THOMAS UHRIK

B.S., Texas A & M University, 1980
B.S., Texas A & M University, 1981

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1986

September, 1986

ISG 86-23

Urbana, Illinois

ABSTRACT

The author gives a rough theory of the use of plausibility functions and evaluation schemes to model plausible reasoning on imprecise, inexact, and incomplete data. A pragmatically motivated methodology for establishing plausibility functions and evaluation schemes in new rule bases is proposed as well as several figures of merit. The techniques presented in the paper are applied to a case study involving soybean diagnosis and monkey behavior discrimination using a rule exerciser which was based on a proposed methodology. The results of the study are summarized and an evaluation of the proposed methods is presented.

ACKNOWLEDGEMENTS

I would like to thank my thesis advisor, Dr. Ryszard Michalski, for countless ideas and suggestions. He provided me with motivation and the original impetus for this thesis. I thank him for the opportunity to set my own goals and approaches to the basic problems.

I also thank Dr. A.B. Baskin, Dr. R. Stepp, Tom Channic, Steve Borodkin, Tony Nowicki, and Bob Reinke, who were also helpful.

I am grateful to several funding agencies which supported my activities: the Office of Naval Research under grant N000 14-82-K-0186, the Defense Advanced Research Project Agency under grant N000 14-K-85-0878, the National Science Foundation under grant MCS 82-05166 and DCR 84-06801, and the United States Department of Agriculture under grant No. 321512344.

Especially, I would also like to thank the Artificial Intelligence Laboratory and the Intelligent Systems Group at the University of Illinois for the use of computers, the learning algorithm GEM, and text processing resources.

TABLE OF CONTENTS

CHAPTER

CHAPTER 1

INTRODUCTION

1.1. Motivation

For many years, expert system designers have been plagued by the problem of how to handle uncertainty and to propagate varying levels of confidence for evidence. Accepted practice among implementors is to adopt a convention for combining various degrees of belief, following a set of general guidelines set by early pioneers in the field of knowledge engineering [ChMi 80, Du 79, Shaf 76]. These early researchers proposed a set of *ad hoc* connectives for plausibility combination, but certain mathematical principles were underlying this choice. In recent years, researchers strove to unify and strengthen these past efforts, remedying previously noted deficiencies [GoNg 85], but the problem is still not resolved.

Individual experimenters quickly found that, depending on the particular problem domain, one set of connectives might be better than another. This non-uniformity of connectives greatly disappointed those in the artificial intelligence community who had hoped for general purpose problem solving engines. A strong trend developed towards domain dependence in AI. Many heuristics cropped up to encapsulate the connective definition that worked best for general classes of problems. For example, for independent plausibilities, the probabilistic sum (PSUM(a,b) = a+b-ab) operator for the OR connective and the PRODUCT operator for the AND connective encompass the plausibility for combined evidence. Yet, the MAXIMUM operator for OR and a linear weighting (typically AVERAGE) for AND might apply to inter-dependent conditions. Set theory and probability and statistics are often used to justify many selections of logical connectives, but because people rarely state their assumptions, it is never quite clear exactly to what extent these various justifications apply.

Many sources for implausibility further cloud the issues. The concepts of uncertainty, imprecision, and inaccuracy, even when restricted to the scope of surface data, are frequently (erroneously) perceived as equivalent bases of plausibility. From the model used in a machine, come other flavors of plausibility, including propagation of surface uncertainty effects, indeterminism in control programs, incomplete data treatment, unreliability of algorithms or hardware executing them, and results of inference processes or application of imprecise rules. In production-based systems, the logical operators are intimately bound to the plausibilities of all constituent terms, but this binding is based upon a rarely stated assumption (i.e., that logical expressions representing rules are rarely *brittle*, meaning that when the reliability of a logical subexpression comes slightly into question, the reliability of expressions including this subexpression do not become completely unreliable but change in a manner consistent with the structure of the expression involved).

A paradigmatic question arises: *so, what does all this mean to the user - what do the delivered plausibilities (numbers) allow the user to conclude?* Some would say - it is a probability - but it is not. Some would say - it is a confidence interval - but it is not. Some would even claim that somehow it measures the sensitivity or insensitivity of the result with respect to small perturbations in surface data (i.e., the lower the reliability of an output, the more likely that an error in the input will cause an output error). The only real answer is that the system is beyond the realm of the new user experiences , and consequently the user must learn the significance of the confidence measures emitted by the analytic device.

Yet, this inherent burden on the new user does not detract from the usefulness of this instrument. Similarly, no one has an innate feel for the use of an analytic balance or mass spectrometer when first exposed to it, but this fact in no way makes the machine disfunctional. Part of the learning process presumes a gradual sensing of the limitations of the instrument. One knows that although an analytical balance can measure a 100g mass, one does not blindly dump the total weight on the platter without careful preliminary adjustment. The limits of the use of plausibility involve both the manner and subject of application. Validity is questionable whenever plausibility is treated as some self-sufficient, unqualifiable

statistic.

The author's purpose here is not to derive an all-encompassing theory of plausibility in expert systems, but rather to emphasise a methodology that one might attempt to apply in designing a system for a new problem domain. The author will try to emphasize the sort of assumptions that are clear to define and to tract as one implements, particularly as this applies to the paraphrastic or explanation facilities the program should present to the user.

Usually, *meta-information* (background information or information about information) accompanies the information sought, but as mentioned before, the interpretation of meta-information also requires a certain measure of experience tempered by wisdom. This paper project this philosophical notion as much as possible.

1.2. Past Formulations of Uncertainty and Plausibility Models

A popular goal of expert systems experts is the ideal of a *calculus of uncertainty*. Whether based on probability theory or logic, a belief is that one day all AI people will be able to sit down together and say "let us calculate the answers to questions". A similar idea was associated with symbolic logic at the time of G. W. Leibnitz [Kr 68]. Although recent efforts have initiated research into a unified theory of the various calculi of uncertainty [GoNg 85], the problem remains open. The most justifiable view at present would be one based upon subjective tenets that change according to problem domain applications. There is some justification for this to be found in measure and confirmation theory. A brief survey of a few techniques are presented.

One interpretation of confidence, in a plausible reasoning sense, is as mathematical probabilities. However, determining that a confidence represents a probability does not say very much, for a probability must be interpreted just as logic variables in rules must be interpreted to have meaning.

Input confidences relate to the notions of imprecise, inaccurate, or incomplete data. If one of these is propagated, the output confidence has an analogous interpretation to that of the input (e.g., if input confidence is the probability of a specified variable being in error, then output confidence might be the probability of a diagnosis being wrong). This idea is simple, but it has the advantage of being well defined. If we try to adopt a hybridized interpretation of input confidence, one gets into trouble unless probability vectors (3 or 4-valued) are used.

Consider the possible internal probabilities that may arise. One can assume equally probable values of variables or use the topology of the space by applying restriction rules to variable values to create probabilities for variable values or conjunctions of values. Degrees of belief about the rules or inference mechanisms being employed (e.g., a conclusion may be based upon a rule that is heuristic and not valid in all cases) provide another source of internal probabilities.

Shortliffe and Buchanan [ShBu 75] attack the source of degrees of belief arising from input confidences and incomplete data (see section 4.5). Hypothesis formation under a Bayesian formalism purports to do no more than resolve uncertainty arising from input credibility and generalization strengths. The lack of total success is due to several reasons: decision classes are not always disjoint, inputs and conditions are not always independent or uncorrelated, and probability distributions are not always completely known. An attractive feature of their scheme is the simultaneous maintenance of belief certainty, and a measure of disbelief. Both of these factors were nebulous and have no surface meaning until they are combined by function giving *degree of belief.*

Duda, Hart and Nilsson [Du 79] also developed a Bayesian model where probabilities are subjectively determined. Here interdependencies in the rule network are accounted for by slight modifications.

Many previous domains of application have adopted the scalar, probabilisticly motivated representation of certainty. The methods employed to handle incomplete and imprecise data is quite intuitive and ad hoc.

Shaffer[Shaf 76] suggests the limitations of a single valued probability, and he develops the idea of upper and lower probabilities. He shows how these ideas apply to the estimation of the confidence for a conclusion, given a premise. Again, these internal measures are solely comparative and subjective.

Similarly, Dempster–Shafer theory [Ga 81] and its implementors adopt an approach representing confidences and margins of error on those confidences. The particular advantage of this approach over Bayesian is the simplicity of application and lack of requirement for substantially complete probability values.

Work in decision making hints that subjective understanding and resolution of uncertainty is quite different from probability theory and Bayes' Rule. This motivated Zadeh to introduce the Fuzzy Set [Za 78] concept and Possibility Theory [Za 79]. The claim is that if we work on a semantic rather than syntactic basis, we can manipulate uncertainty more successfully. Fussy Set calculus operates over linguistic rather than numeric terms. Zadeh claims to have covered broader problems than many (partial matching of antecedents).

Hybrid systems, such as DISCIPLE [To 82], INFERNO [Qu 82], ERIS [Fr 80, Fr 81] and ADVISE [MiBa 84], attempt to integrate many of these ideas, but allowing the end user to experiment extensively. In this spirit, the ideas presented herein were developed.

CHAPTER 2

APPROXIMATE REASONING THROUGH PLAUSIBILITY FUNCTIONS

2.1. Evaluation and Propagation of Uncertainty in Condition–Action Rules

In this chapter, the author illustrates a rationale behind choosing a plausibility or confidence folding function in a rule based *production system*. From there, the horisons expand.

Consider the simple production rule :

$$A \,\&\, B \rightarrow C \ .$$

Humans seem to interpret this sort of rule intuitively and consistently in the absence of further information. For the established absolute certainty of conditions A and B, C should obviously be established with equal certainty. Yet, the absolute failure of either A or B does not preclude the possibility of C occurring via some other production or perhaps stochastically (ie., not causal relative to A or B). People naturally make allowances for such unanticipated causalities and events by accepting them as a characteristic of physical reality. Hence, in a rule system where goal variables are based upon uncertain input variables and subsequent uncertain intermediate variables, the proper treatment of uncertainty can serve as a mechanism for inexact reasoning.

This approximate reasoning can be modeled by maintaining a certainty value (analogous to a probability) for each variable or condition in the rules and imposing a scheme for propagating certainties across logical operators of rules. For instance, if one assigns *confidence* values (in the range 0–1) to the truth of A and B in the rule above, the MIN function might be assigned to the logical "&" operator, so that the confidence of C is systematically determined in the course of the logical execution of the rule:

$$\text{confidence}(C) = \text{MIN} \ (\ \text{confidence}(A) \ , \ \text{confidence}(B) \ ) \ .$$

The absolute magnitude of such certainty values need have no real significance, but usually only its relative magnitude with respect to other certainties is important. This mechanism employs a set of

functions, corresponding to the logical operators in rules, for evaluating the combined certainty of variables or conditions (subexpressions). Given a function f which is a member of this set of functions and values c_A (confidence of A), c_B (confidence of B), $c_{A'}$ (confidence of *not* A), $c_{B'}$ (confidence of *not* B), δ_1 (some small number), δ_2 (another small number), f_{max} (the maximum value of f), k (a convenient constant), then some desirable features function f and its generalized (variable arity) functional are :

1) local increasing monotonicity, i.e., that within an arbitrary neighborhood of a point, the function should be increasing with positive increase in any component certainty values,
$$f(c_A,c_B) \leq f(c_A+\delta_1,c_B),\ f(c_A,c_B) \leq f(c_A,c_B+\delta_2)\ , \qquad \text{for any } \delta_1,\delta_2\ ,$$
2) complementation results in inverted certainties,
$$f(c_A,c_B)/f_{max} \leq 1 - k\times f(c_{A'},c_{B'})\ ,$$
3) the functional be homologous or piecewise homologous, for the purposes of possible future mathematical analysis,
4) the function is easily evaluated so as not to require too much computing time since it is common for confidence values to be continuously reevaluated throughout the course of an expert system consultation,
5) the functional be paraphrasable since it is necessary for expert systems to have at least some minimal capability to explain to a human user how it arrived at certain conclusions or decisions,
6) the functional have a recursive definition that corresponds to the recursive structure of the subformulas of the logic system employed, partly to ensure 4) but also to aid in the implementation in a standard programming language.

It is noted that the evaluation of the truth values in the logic and the confidence values in the plausibility system are generally two different concepts. Any semantic binding between the two concepts is clearly an assumption or definition in an implementation.

By way of example, consider an information processor (presumably human) for the CIA. He receives information from 2 field agents, X and Y. X is correct 70% of the time in his information, and source Y is correct 90% of the time. Suppose that *source X reports the Soviet Union will adjust the exchange ruble 5% in favor of Bulgaria*, and *source Y reports the Soviet Union has increased cultural exchange with Bulgaria.* If there is a general policy directive that *if the Soviet Union increases the exchange ruble in favor of Bulgaria AND also increases cultural exchange then an investigation of a possible Bulgarian covert operation is indicated.* In order to apply the rule, the evidence of the first fact, having confidence of 0.7, must be combined via an AND operator with evidence of the second fact, having confidence of 0.9 . Here, the result

is f(0.7,0.9), the value of f_{max} is 1.0, and the principle of 1) indicates that if the confidence of either evidence increases then the resulting confidence for the *AND* condition is increased. Principle 2) suggests that the confidence in the statement *cultural exchange between the Soviet Union and Bulgaria has not increased* is less than or equal 10%.

Given input confidences c_1, c_2, ... , c_n, some useful functions that are used in assigning evidence to a conclusion drawn from the facts relating to these confidences are:

PROD - the arithmetic product, $PROD(c_1,c_2,...) = \Pi\, c_i$,

MIN - minimum value of arguments,

AVG - arithmetic mean, $AVG(c_1,c_2,...) = (c_1+c_2)/2$,

GEOM - geometric mean, $GEOM(c_1,c_2) = \sqrt{c_1c_2}$,

MAX - maximum value of arguments,

PSUM - probabilistic sum, $PSUM(c_1,c_2,...) = c_1+c_2-c_1c_2$,

RMS - root-mean-square (from physics and engineering), $RMS(c_1,c_2,...) = \sqrt{(c_1^2+c_2^2+...)/n}$,

RMP - root-mean-power, given a constant p, $RMP(c_1,c_2,...) = \sqrt[p]{(c_1^p+c_2^p+...)/n}$

or some more esoteric functions such as :

RSR - reciprocal of sum of reciprocals (from reliability engineering), $1/\Sigma(1/c_i)$,

N*RSR - n times RSR, $n/\Sigma(1/c_i)$

NPRSRP- n to a power, p, over reciprocals to the power, p, $n^p/\Sigma(1/c_i)^p$

ISDP - iterative sum - divide by some constant, P,

$$((...(c_1+c_2)/p)+c_3)/p)+c_4)/p)+...)/p+c_n) = (\Sigma\, c_i/P^{n-i+1}) + (1-p)c_1/P^n,$$

and many others serve in their own right as is illustrated in the next section. Note that some of these functions require additional modifications to satisfy the constraints of the postulates. GEOM and ISDP are not invariant with respect to the order of recursive application. This can be remedied by specifying an order constraint on recursive use. Denote this by

GEOMSF - recursively apply GEOM to smallest arguments first ,

GEOMBF - recursively apply GEOM to biggest arguments first ,

ISDPSF - recursively apply ISDP to smallest arguments first ,

ISDPBF - recursively apply ISDP to biggest arguments first .

The biggest-to-smallest and smallest-to-biggest represents the most useful variations of ordering since the results of other orderings fall between these (refer to section 3.2 in Chapter 3). Yet, for GEOM. the problem might also be solved by considering a generalization of GEOM that is identical to the classical geometric mean for two terms. The motivation for this classical notion is: given a rectangle with sides a,b , find a square with the same area. The side of such a square will be the geometric mean of a and b. So generalising area to volume and the rectangle to a right-parallelepiped, the task is to find a cube with equivalent volume. Further, one extends to n-space, finding that the rectangle becomes a rectilinear simplex so the formula becomes

$$GEOM'(c_1, \ldots, c_n) = \left(\prod c_i \right)^{1/n} .$$

Alternatively, one might think of some extensions of "rectangle" such as symmetric polygon, etc., but one runs into problems with these geometric generalizations and consequently abandons the idea. Looking at the formula for GEOM, leads one to propose other generalisations :

$$GEOM''(\ldots) = \left(\prod c_i \right)^{1/2}$$

$$GEOM'''(c_1, \ldots, c_i) = \left(c_i \cdot GEOM'''(c_1, \ldots, c_i) \right)^{1/i}$$

with $GEOM'''(c_1,c_2)=GEOM(c_1,c_2)$ and an ordering constraint,

on $c_1 , c_2 , c_3 , \ldots , c_i$,

but these are not estheticly pleasing.

As noted earlier, the trend has been to choose one of the standard default functions such as MIN/MAX for AND/OR combinations. If the problem domain appears somewhat sticky, the only solution seems to be to run a costly series of experiments to discover the function that works best. Rather than

proceeding blindly, one can be more clever with a few of these functions by arranging them into groups sharing a number of properties in common. One needs only to try the most characteristic functions and the more detailed experiments can concentrate on functions similar to the most promising.

Towards this end, note that a partial ordering exists over these functions (see Appendix A – for details), such that,

$$0 \leq \ldots \leq \text{PROD} \leq \text{MIN} \leq \text{AVE} \leq \text{MAX} \leq \text{PSUM} \leq \ldots \leq 1$$

and thus there exists a spectrum of functions which combine uncertainties together. Ryszard Michalski suggests that one can derive a function which possesses the qualities desired by taking linear combinations of the above functions. More specifically, combinations containing a parameter α:

$$\alpha f_1(a,b) + (1-\alpha)f_2(a,b),$$

such as,

$$\alpha\text{MIN}(a,b) + (1-\alpha)\text{MAX}(a,b)$$

or

$$\alpha\text{PROD}(a,b) + (1-\alpha)\text{PSUM}(a,b) \quad ,$$

provide the ability to span the spectrum by adjusting the parameter over the interval $[0,1]$. This function captures several functions in one spectrum so that experiments on that spectrum will summarize information for intervals in that spectrum. Rather than testing MIN, MAX, AVE separately, one has discovered a function that will tell something about each. In general, a methodology might involve testing for the effects of a function at the low end of the spectrum compared against a specimen from the high end of the spectrum. The results of in-between functions could be "interpolated". Furthermore, this idea extends to a more general notion of a row vector of functions $\mathbf{F}$ multiplied by a column vector of weights or parameters $\hat{\alpha}$ yields

$$\hat{\alpha}\mathbf{F} = \Sigma\alpha_i f_i \quad .$$

2.2. Examples of Specific Domain Types

In order to illustrate some of the techniques given, the author proposes to attack the question of plausibility functions for the use in the Soybean Disease Diagnosis domain. This domain has prime examples of quasi-defined objects that arise in expert system development. The problem will be further delineated later in this chapter and in Chapter 8. In general, the research effort tried was to effect uncertainty propagation through terms, complexes, and rules using the individual functions as well as the spectral function (with the simple, single parameter method mentioned in 2.1). The preliminary strategy is to use the PROD-AVG combination for & ("AND") and the PSUM-AVG for V ("OR") since the former favors dependence separation (the lower spectrum - including MIN) and the latter favors mutual exclusive separation (the upper spectrum - including MAX).

Please note that not all problem domains have clouded ideas about plausibility. In fact, a number of domains maintain rigid criteria for the propagation of confidences and plausibilities, so for these domains, no empirical work on plausibilty is needed. Rather, the definitions of domain experts should be adhered to.

One such domain is system reliability analysis (a subfield of industrial engineering). Here, quality assurance experts have invested considerable time and effort to construct models of decay, failure processes, and the effect upon the total of a system as a consequence of its components.

Most systems are decomposable into serial elements, for which failure of any component in the element prevents the continued reliable performance of the element, and parallel elements, for which only the simultaneous failure of all components impairs the reliable performance of the element. For example, consider the system depicted in Figure 1. The general formulas for the holistic reliability of the system in Figure 1 are the recursive formulas for parallel and serial reliability combination:

$$R_{serial} = \Pi R_i ,$$

$$R_{parallel} = 1 - \Pi (1 - R_i) ,$$

where R_{serial} is the reliability rating of an arbitrary serial configuration, $R_{parallel}$ is the reliability rating of

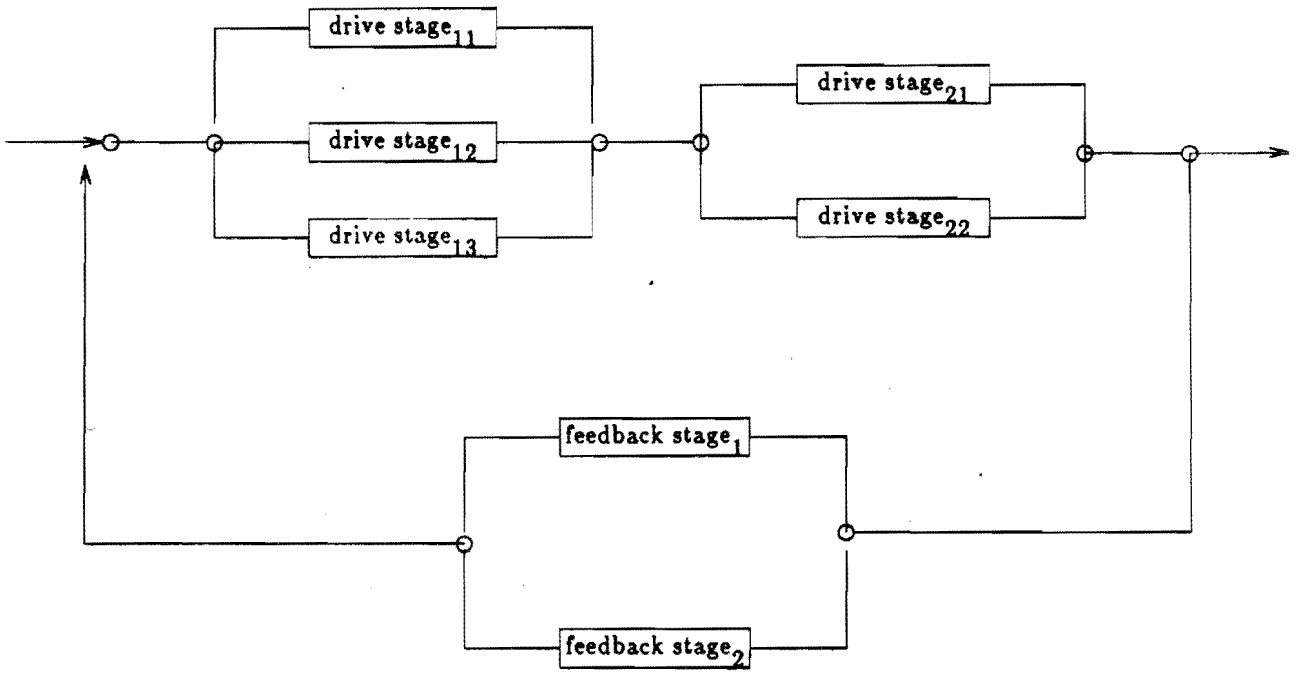


Figure 1. A System of Serial and Parallel Elements.

an arbitrary parallel configuration, and R_i is the reliability rating of component i in the system. Note that the parallel formula reverts to a simple PSUM for the case of two component reliabilities. These formulas are analogous to the algorithm for obtaining the effective impedance in an electrical network in terms of the component impedances . Actually, the reliabilities in this model describe time dynamic probability of the system continuing to function, and they are defined in terms of failure rates for Poisson distributions ($R_i = e^{-\lambda \Delta t}$ is the reliability or probability that no failure occurs after time Δt, $1 - e^{-\lambda \Delta t}$ is probability of failure after time Δt, λ is the failure rate $= 1/m$, and m is the mean time to failure) [Am 71]. The R_{serial} and $R_{parallel}$ expressions can be reformulated in terms of mean time to failure for serial and parallel configurations:

$$m_{serial} = 1/(\Sigma\ 1/m_i)$$

$$m_{parallel} = \Sigma\ m_i - \Sigma\ 1/(1/m_i+m_j) + \Sigma\ 1/(1/m_i+m_j+m_k) - \ldots\ .$$

The network in Figure 1 might also be represented as a rule set for calculating the reliability of a system and logical side effects of component failure. The AND and OR functions of these rules could be given the evaluation patterns of the parallel and serial reliability functions given above. In that case, the rules might be as in Figure 2. Any change to the network would require a change in this rule structure.

This sort of application is highly desirable, but unfortunately, not enough experts have come together to decide the criteria that they will mutually agree to adopt for discussing plausibility in their domain. Hence, a motivation for the remainder of this paper exists.

|DRIVE_STAGE_I = on| |DRIVE_STAGE_II = on| |FEEDBACK_STAGE = on|
$\rightarrow$
|SYSTEM = on|

|DRIVER11 = on| V |DRIVER12 = on| V |DRIVER13 = on|
$\rightarrow$
|DRIVE_STAGE_I = on|

|DRIVER21 = on| V |DRIVER22 = on|
$\rightarrow$
|DRIVE_STAGE_II = on|

|FEEDBACK1 = on| V |FEEDBACK2 = on|
$\rightarrow$
|FEEDBACK_STAGE = on|

Figure 2. Set of Rules for Calculating Reliability of Serial and Parallel Elements.

2.3. Decision Rules and Diagnosis

The particular class of problems adopted for this research effort is that of diagnosis through decision rules. This combines many aspects of deductive reasoning with a need for flexible plausibility propagation. At this time, some terms, restrictions, and paradigmatic conceptions of problems with decision rules are illustrated.

First of all, the author introduces the idea of a decision rule and its components. A decision rule is a fairly intuitive concept, but we define a particular use for clarity. As stated earlier, a rule will be of the form:

$$\text{LHS} \rightarrow \text{RHS}$$

where LHS (the left hand side) represents some testable condition which is *usually* composed of logic formulas, RHS (the right hand side) represents an executable action, goal, or satisfiable assertion. Given that the LHS is *satisfied*, then the RHS is *interpreted*. The term "satisfied" will be defined later. Let us say that satisfaction returns a value. The interpretation consists of instantiating variables in the RHS to TRUE or FALSE or some other values (possibly executing some action as a side effect) and attaching confidence values to the RHS target variables (the variables instantiated to TRUE or FALSE or other). Some restrictions made for the purpose of further illustrations follow:

1) the LHS will be comprised of VL_1 formulas [Mi 74],

2) the RHS will be simple TRUE/FALSE assignment statements,

3) the rules will be assumed to be pertinent to a classification problem
and the targets of the right hand sides will be "classes",

4) the use of plausibilities is assumed appropriate to the domain at hand .

Furthermore, the term *decision rule* designates the idea of approximate satisfaction of rules with some stated value of confidence denoting the degree to which a LHS is *partially satisfied*. This will be denoted explicitly as:

$$\text{LHS} \rightarrow \text{RHS} \qquad \text{(confidence)}$$

where confidence is a real number in the interval [0,1]. In general, the terms "rule", "decision rule", ... will be used interchangeablely because there is no purpose in referring to the strict, unextended notion of a production rule in its most primitive form and control mechanism.

Indeed, the plausibility notion is a rather complex manifestation of control scheme concerns. The sections that follow define some terminology for later use and describe the structure of rules used in the context of examples taken from the soybean diagnosis problem (called "PLANT" for short). Therein, some need for the ultimate complexity of these rules (particularly the *weights*) is indicated by the discussion of the issues similar to the ones human experts are required to go through in validating expert systems.

2.3.1. Structure of VL_1 Rules

Previous papers discuss at great length the syntax and use of VL_1 formulas [Mi 74, Ch 79]. A brief description is included here.

A *selector* is the basic unit of logical truth in the VL formalism. The abstract format is

$$[x_i \# R]$$

where x_i is variable which takes on values from domain D_i, # is a relation from among $\{ =, \neq, <, >, \leq, \geq \}$, and $R \subseteq D_i$ is the *reference set* for the selector, specifying values to be tested against the variable's value. For the convenience of specifying a confidence value for the selector in addition to its truth value, a modified selector

$$[x_i \# R : w]$$

is introduced, where w is a weight function which specifies the confidence value for various values in the reference set.

A concatenation of selectors form a *complex*. Complexes implicitly contain the logical AND operator (&) between selectors, but for clearity, complexes split across lines always make the operator explicit. For example,

$$[X = 5][Y = 6][Color = red \text{ V } blue]$$

$$\& \; [A \geq 10 \,]$$

is a complex indicating the logical conjunction of four selectors.

A *disjunctive term* is a disjunction of complexes, and a *linear expression* is a linear combination of disjunctive terms. Thus,

$$0.8\ [X = 2][Y > 5] \text{ V } [X = 1][Y > 10] + 0.2\ [X > 2][Y = 4] \text{ V } [Z = 3][Y < 4]$$

is a linear expression consisting of two disjunctive terms, each consisting of two complexes, each of which contains two selectors.

This terminology and that presented in the next section is summarised in Figure 3.

2.3.2. Satisfaction and Partial/Approximate Satisfaction

A rule is said to be *satisfied* when some of the user input data matches all of the *selectors* in the rule. A *selector* is the basic unit of a rule which tests truth of the condition that the selector variable's

Syntax		Assigned Confidence Value
Selector S:	$[x_i \# R : w]$	1 if $x_i \in R$, 0 if $x_i \notin R$, 0.5 if unknown
Complex C:	$S_1 S_2 S_3 \ldots$	AND of selector confidence values
Disjunctive Term DT:	$C_1 \text{ V } C_2 \text{ V } C_3 \ldots$	OR of confidence values of complexes
Linear Expression:	$q_1 DT_1 + q_2 DT_2 \ldots$	arithmetic combination of disjunctive term confidences

*Note: $\# \in \{ =, \neq, <, >, \geq, \leq \}$ denotes a relational operator.
R is a subset of x_i's value set.

Figure 3. Structure and Evaluation Schemes of Rules.

value satisfies the relation for some value in the reference set. In fact, this can be expressed as a disjunction of tests over all values of the reference set, and hence selectors have the form

$$[x_i \# val_1 \vee val_2 \vee val_3 \vee \ldots] .$$

This *internal disjunction* of selector truth values yield both the selector truth value and confidence. These combine via an AND function to give the complex truth and confidence, and subsequently an OR function is applied across disjunctive terms. Coefficients are multiplied by the confidence values of disjunctive terms and summed to give the confidence of a linear expression.

Rules can be composed of either a selector, a complex, a disjunctive term, or a linear expression. Thus, a rule

$$[x_2 = 3][x_3 = 1 \vee 3] \vee [x_4 < 4] \rightarrow [\text{decision} = A]$$

can be interpreted: Decision A is taken if x_2 is 3 and x_3 is 1 or 3, or if x_4 is smaller than 4.

In a case of perfect knowledge of symptom descriptors, there is little doubt that the rules are more than adequate to discriminate various cases of the diseases. Yet, in most situations, sparse information on the state of input variables requires efficient use the little data available. Hence, an *evaluation scheme* is defined to handle the numerous cases of *partial satisfaction* of a group of selectors in a rule. Such a scheme includes the assignment of confidence values to the basic conditional units, the semantics of the AND and OR functions (eg., MIN/MAX), and the subsequent assignment of confidences to right-hand sides of rules. Figure 3 describes the structure and evaluation schema for rules.

2.3.3. Complications in Rules and their Structure

Some rules require complex structures to capture the complexity of analysis and proper weighting of components of information that human experts deliver to such problem as soybean disease diagnosis (PLANT [ChMi 80]). Some details of this particular domain are elucidated in this section.

In the soybean domain, the greatest confusion occurs among the various leaf spot diseases (which are the most numerous among the various diseases) or involve instances like 3 leaf spot diseases and

another disease. A superficial look at the rules might lead one to conclude that since the leaf spot diseases have the longest rules (number of selectors), a high potential for confusion is expected in the case of partial satisfaction even without regard for any understanding of the deep nature of the problem. Consider the following rules:

[leafspots halos=pres][seed mold growth=pres]
[leaf mildew growth=on lower leaf surface][stem=norm]
→ [diagnosis=downy mildew]

and

[precipitation>norm][# yrs crop repeated>1]
[damaged area = not whole field][roots=n]
[leaves=abn][leafspots halos=no yellow halos]
[leafspots water soaked margin=abs]
[leafspot size>1/8"][leaf malformation=abs]
V [precipitation>norm][leaves=abn][leafspot size>1/8"]
[leafspots halos=no yellow halos]
[leafspots watersoaked margin = abs][roots=norm]
V [time=apr..jun][damaged area=not whole field]
[leaves=abn][leafspot halos=no yellow halos]
[leafspots water soaked margin=abs][roots=norm]
→ [diagnosis=brown spot]

are the rules for downy mildew and brown spot. Both of these rules adequately cover the standard textbook cases of the two diseases, but consider the event that one knows that leafspot halos exist with no yellow halos and that stem and roots are normal, leaf mildew growth is observed, water soaked margins are absent, and leaf spot size is greater than 1/8". Further, from the given information about leaves, it is inferred that leaves are abnormal (either by structural relations among variables or explicit background knowledge rules). Assuming the evaluation scheme for the PLANT INDUCTION RULES (PSUM/AVE) [Ch 79], results in 3 of 4 selectors satisfied in the first rule, giving 0.75 confidence. Since many of the same selectors occur also in the second rule, the confidence value gotten here must be evaluated also. The first complex is partially satisfied by

[leaves=abn][leafspots halos=no yellow halos][roots=n]

with confidence of 0.33 . The second complex is satisfied by

[leaves=abn][leafspots halos=no yellow halos]
[roots=n][leaf spots > 1/8"]

with a confidence of 0.66 , and the third is satisfied by

[leaves=abn][leafspots halos=no yellow halos]
[roots=n][leaf spots water soaked margin=abs]

also with 0.66, which brings the cumulative confidence to 0.93 . Thus, although there is good evidence for downy mildew, brown spot is indicated as the most probable choice on the basis of inflated confidence due to the common sub–complex,

[leaves=abn][leafspots halos=no yellow halos][roots=n]

which is a substantial component of each complex. Granted, if one selects the threshold low enough, one can empirically guarantee for large test sets that the correct diagnosis of the disease will always be indicated, but this only increases the confusion between similar diseases when this is done. This lead to the problem that soybean diagnosis was only trivially solved for easiest cases (the cases the machine readily discriminates are no different than those classics previously noted by soy pathologists).

Problems in the above example arise from several different sources. One consideration is the interplay of values between decision rules and background rules. PLANT contains knowledge in the form of structures of variables and inherited values. Note the explicit satisfaction of

[leafspot halos=no yellow halos]

and the default satisfaction of

[leaves=abnormal]

based upon the inference that was assumed based upon the values of variables pertinent to leaves. Use of structured variables in a complex may not take account of the commonalities this sets up with other complexes. There is a need to distinguish between the sources of confidence, so that such inflationary combination does not occur. Satisfaction may be context sensitive in many cases. (Well, maybe even humans cannot isolate these contexts very well ?)

This leads to a more general question of how commonalities between rules should be treated in the evaluation scheme. Groups of selectors can occur in clusters in more than one term in the same rule. The simultaneous satisfaction of such a group, as

[roots=n][leaves=abn][leafspot halos=no yellow halos]

in the example above, allows this one condition to artificially inflate the significance of the rule by raising the confidence value out of proportion. It would seem that these redundant complexes should be factored out into a separate rule. Often, this indicates a significant generalization on its own and can be assigned a meaningful name, as with the complex common to Brown Spot, Frog Eye Leaf Spot, Alternaria Leaf Spot, and Phyllosticta Leaf Spot :

[leaves=abn][leafspot halos=no yellowhalos]
[leafspots watersoaked margin =abs]
[leafspot size>1/8"]
→ leafspot disease

where leafspot disease can serve within a selector with the proper weight in the contexts of the previous occurrence of the cluster. Several of these factorings of complexes have been noted in Figure 4 for both expert and induction rules.

Another problem exists in the homogeneity of the evaluation scheme. The leafspot diseases are simply too similar to assume that the same scheme for distinguishing the most diverse diseases will also distinguish the fine shades of the leaf diseases. There is a need for different evaluation schemes to evaluate different rules either staticly as expressed in different layers of control within a rule or dynamically. This may even be dependent on the topology of rules as well (e.g., f(# of terms)).

The average function may serve well as a default since it has been empirically proven the best homogeneous scheme, but the minimum function, which yields a stricter separation of confidences, suits the leafspot disease rules better. The switching between various levels of rules might be effected through the use of explicitly separated rule groups as in the ADVISE system [MiBa 84, Uh 82].

Induction Rules

[stem=abn][leaf malformed=abs]
→ stem disease

[stem=abn][leaf=abn]
→ stem & leaf disease

[leaves=abn][leafspot=pres]
[leaf spot size>1/8"]
[leafspot water soaked margin=abs]
→ leafcond1

[roots=norm][prec=>norm]
[leafcond1]
→ watercondition

[leafspot water soaked margin=pres]
[leafspot halos=not yellow]
→ leafcond2

[leaves=abn][leafspot size>1/8"]
→ leafcond3

[leafcondition3][stem=n]
→ leafcond4

Expert Rules

[leaves=abn][stem=abn]
→ extensive damage

[plant stand<normal][extensive damage]
[roots=rotted][stem cankers=present]
[canker lesion color=brown,black]
→ root rot

[leaves=abn][leafspot halos=not yellow]
[prec=>norm]
→ halo syndrome

[leaves=abn][leafspot halos=not yellow]
[leafspot water soaked margin=abs]
[leafspot size>1/8"]
→ leafspot disease

Figure 4 Partial Term Factors Occurring in Multiple Terms/Rules

The intersection (AND) of the exclusion sets of the selectors in a rule group may serve to define the left-hand side of a rule for switching rule groups. As soon as the confidence of that rule surpasses all the current active rule group, the new rule group would be activated (and optionally deactivate the other). See Figure 5 for an illustration of this rule group switching.

The probabilistic sum used in the induction evaluation is certainly justified in rules where the number of terms and complexes are few and distinct, but the leafspot diseases indicate the flaw with trying to apply the technique too loosely. If there is only marginal evidence in a great many complexes, we have suddenly inflated the significance of these cumulatively (in fact they probably have no significance when considered en masse). Since there is clearly a variant need, dependent on the rule, as to the choice of evaluation scheme, the scheme should be flexible in this respect.

The following is a rule group
to distinguish between mildew diseases:

Rule1
[leaves=abn][leaf malformation=abs]
[leaf mildew growth=on upper suface][roots=norm]
⟶
[Diagnosis=Powdery mildew]

Rule2
[leafspot halos=pres]
[stem=norm][seed mold growth=pres]
[leaf mildew growth=on lower leaf surface]
⟶
[Diagnosis=Downy mildew]

The intersection of the exclusion sets of
selectors in these rules (i.e., conjunction
of selectors having all attribute values not
represented in either rule) describes all cases
where neither rule is satisfied and thus a left
hand side of a rule for switching rule groups:

[leaves=norm][roots=abn][seed mold growth=abs]
[stem=abn][leafspot halos=abs]
⟶
[Rule Group≠Mildew Group]

Figure 5 – Formulation of Rule Group Switching Rules

2.4. Red Herrings

The possible uses of plausibility functions are described in the following pages, but one thing that they ought not to be used for is resolving inappropriate or poorly engineered question options presented to the end-system user. Consider the following variable data gathering question to be implemented in an expert system for sociology case studies:

Have you stopped beating your wife yet ?

(1) Yes

(2) No

If the variable in this example is intended to be used in the system for the notion "does/does not beat spouse", it is totally inappropriate as the wording of the question introduces a severe bias which frustrates any sane user. If a confidence is now solicited, the user is very inclined to seek revenge by

specifying a low confidence for whatever interpretation he adopted to ask this complex question. In fact, whenever the options for a variable overlap or exclude some value the user has observed, we stand the chance of introducing this artificial unconfidence.

2.4.1. IMPRECISE / INACCURATE / INCOMPLETE

There is often a distinction between between precision, accuracy, and completeness of data that is confused in the treatment of plausible reasoning. In fact, these may be combined in the last stage of plausibility propagation, but for purposes of paraphrasing results and explanations, this combination of various contributing factors in the TRUTH and FALSITY of any conclusion should be deferred until the very last step of computation [To 82].

Precision of data is a measure of how often a similar value will be delivered in the context of a particular event. This is primarily a quality determined by the style of measurement or instrumentation used to collect data. For instance, sampling rates can affect precision.

Accuracy is often confused with precision. Accuracy is measure of how close to the true value a measurement is. Some instruments are extremely precise, but not particularly accurate. This quality can also be applied to humans. For instance, if Mr. X always misjudges value a in place of value b for some attribute, he is extremely precise (always the same answer) but not very accurate. Usually accuracy is more important, but precision indicates reproducibility of results.

Completeness of data refers to the frequent case when not all the attribute values of an event are known. Human memory is flawed, and the observer often forgets a detail or neglects to write it down. Furthermore, as a session with an expert system proceeds, not all the data is known to the system until the end. The system initially marks all variables as unknown and use some scheme for handling incomplete data to arrive at preliminary results at each stage of the session. This is desirable because often while being asked about one thing, the human user will remember a fact that previously was forgotten, and this fact can be registered.

2.4.2. Internal vs. External Plausibilities

A further distinction should be made between *internal and external* plausibilities. All of the above considerations apply to purely surface data that an expert system might be collecting along with values of variables. Often, an overall plausibility figure may be assigned to the source of the information or the rules. For example, Mr. A may be an unreliable source of information, or Dr. X may be an unreliable source of rules (perhaps rules even come from various sources in the system). This sort of plausibility weighting is usually used in the final steps of evaluation of confidence.

2.4.3. UNKNOWN Values

In addition to the above distinctions, several values of "UNKNOWN" manifest themselves. These are represented separately in the meta–expert system ADVISE [MiBa 84], but in other systems, they are often times treated in a uniform fashion, and hence, semanticly confused. These entities are :

1) UNKNOWN - simple unknown variety,

2) UNDEFINED - this is intrinsicly unknowable,

3) DO NOT CARE - this should satisfy any condition,

4) DOES NOT APPLY - this should fail any condition.

This distinction, although in truth a control consideration, is often assigned plausibility significance. Only 1) applies uniquely to plausibility. The others, 2) through 4), affect the pattern of user input solicitation for a given expert system, e.g., whether attributes with these values are to be requested again indefinitely, just once, or never.

CHAPTER 3

OPERATIONAL CONSIDERATIONS AND INTERNAL MODELS

3.1. Parameterisation of Functionals

For a specific domain or problem, using a linear combination of 2 functions to span a spectrum of candidate functions manages to cut down on the number of separate entities that might be examined in the search for a custom fit to domain dependencies. However, one can not hope to fit all possible functions of interest into such a linear spectrum.

Yet, it is possible to generalise by proceeding in a slightly different direction. Basicly, the desirable properties of an idealised combination function may be retained by introducing a parameter α and arriving at an ordering of the particular functions with respect to the parameter α. In the last section, the parameterization was a simple linear combination with α occurring in the coefficients. A more comprehensive notion is that of a general functional (i.e., a function of arbitrary arity) parameterized in arbitrary ways with respect to a set of parameters. This is overstepping the need since the chief interest is the spectral ordering of the functions. Thus, one parameter or degree of freedom is sufficient to delineate one function from the next in the sequence.

Further, since the exact magnitude of the function at any point is not of concern, a tolerance allows the function to take a range of values that satisfy the ordering criterion of relative magnitudes. For functions of discrete variables, this is a helpful metric and may even be indispensable in determining experimental values. Sometimes a domain expert can give a plausibility judgement for a few select test cases. Usually, this amounts to little more than a boolean decision. To make wide use of this data, a set of criterion can be devised that optimally fit his yes–no answers to an analytic function along with an approximate tolerance function which designates allowable deviations from the analytic function for use in sensitivity analysis.

As a special case, consider a two dimensional plausibility space (perhaps a projection of a homogeneous hyperspace). Let us imagine a view of the plot of a combinate plausibility function in 3-space. This plot is depicted as a surface assuming continuity and appears in Figure 6 for the AVE function. A spectrum of such functions is depicted in a similar fashion in Figure 7. Note the progression of surfaces, one above the other, maintaining the same shape. Note also that the AVE plane falls roughly in the center.

3.2. Different Styles of Parameterisations

A wide variety of parameterisation styles and types exist. Only a few, interesting examples will be presented herein, but observations can be made regarding parameterisations of functionals.

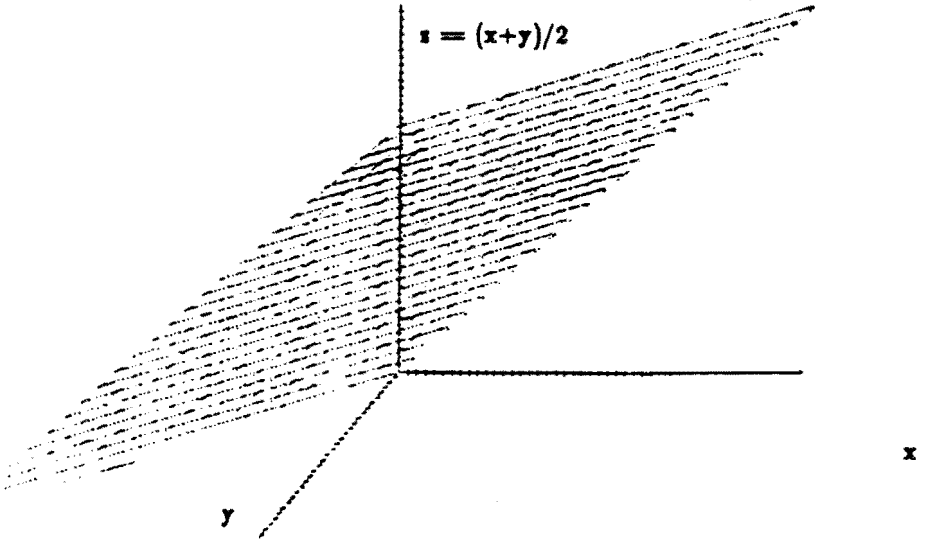


Figure 6. Three Dimensional Plot of the Average Function.

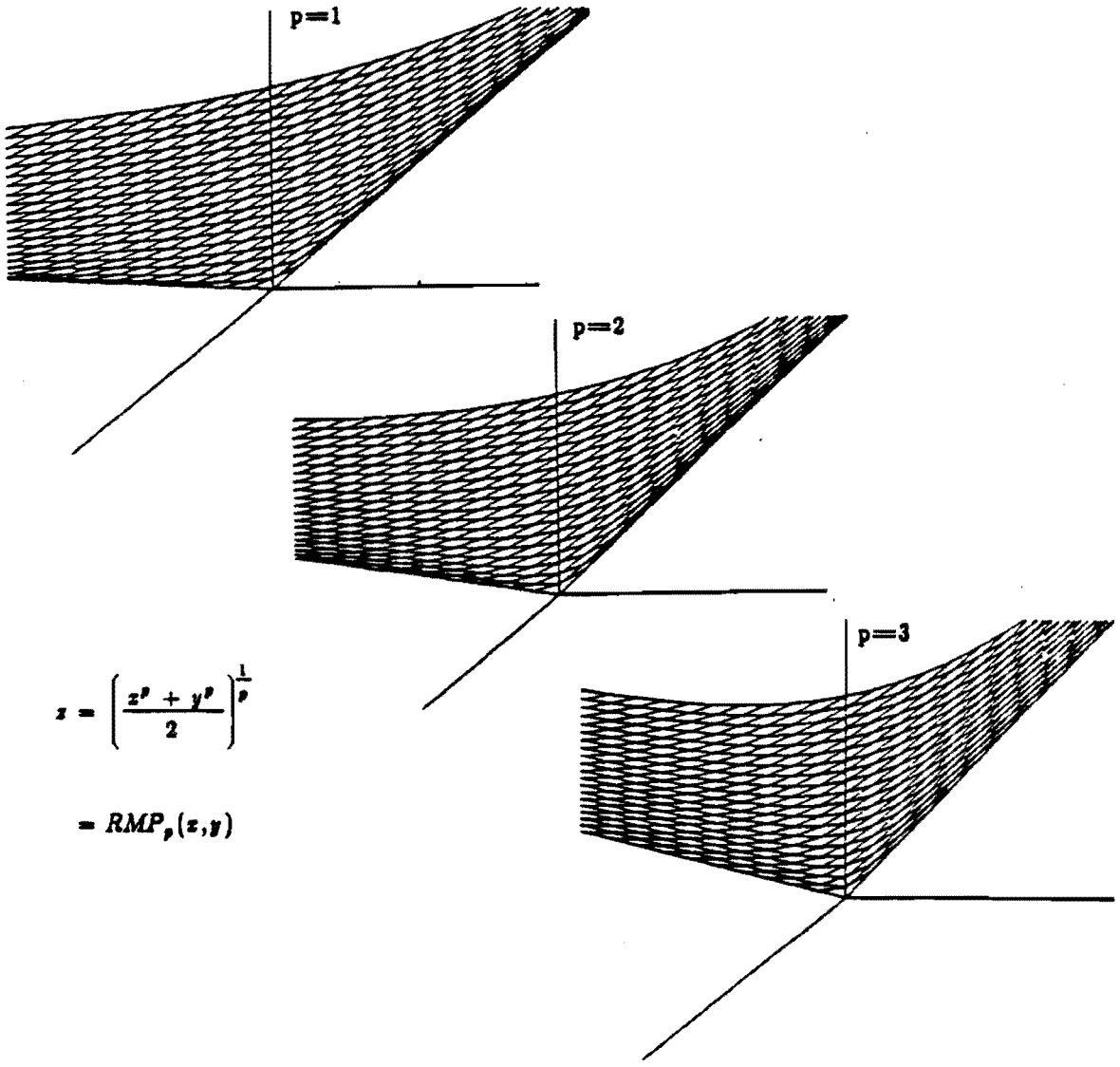


Figure 7. Three Dimensional Plot of a Progression of Plausibility Functions.

The commonest parameterisation type corresponds to a gradual, but uniform, change in function value over a family of curves. E.g. , consider the transistor I_C vs. V_{CE} characteristic curves parameterized with respect to I_{BE}.

The next most common parameterization form is selection among drastically irreconcilable curves. Here, the I–V characteristic curve of a diode serves as a good example: it suffices to divide the curve into 3 or more linear components.

The ISDP style of function and generalized GEOM functions demonstrate that a parameterization of a functional may depend on orderings of the arguments. This has mostly to do with the manner in which the basic function kernel is generalized to a functional (section 2.2.2). In fact for these particular functionals, we can generate a spectrum of functions by simply permuting the order criteria. Thus for for $x_1, \ldots, x_n$, possible orderings might be enumerated:

$$x_1 > x_2 > x_3 > x_4 > \ldots > x_n$$

$$x_2 > x_1 > x_3 > x_4 > \ldots > x_n$$

$$x_2 > x_3 > x_1 > x_4 > \ldots > x_n$$

$$\bullet \bullet \bullet$$

$$x_n > \ldots > x_4 > x_3 > x_2 > x_1$$

with n! possible orderings. Let us denote these as $O_{1234..n}$, $O_{2134..n}$, $O_{2314..n}$, ... , $O_{n..4321}$ corresponding to a lexicographic permutation as above. It is clear (as mentioned earlier) that both ISD2 and GEOM are bounded by the specification of orderings $O_{123..n}$ and $O_{n..321}$. All the GEOM and ISD2 variations with other orderings fall between these two respectively. In general, we might propose the following :

1) GEOM with ordering $O_{a_1a_2\ldots N\ldots M\ldots a_n}$ such that N > M
is less than or equal to
GEOM with ordering $O_{a_1a_2\ldots M\ldots N\ldots a_n}$,

2) ISD2 with ordering $O_{a_1a_2\ldots N\ldots M\ldots a_n}$ such that N > M
is less than or equal to
ISD2 with ordering $O_{a_1a_2\ldots M\ldots N\ldots a_n}$.

It follows inductively that the lexicographic sequence of permutations maintains the lexicographic ordering for any similar functional based on an antisymmetric kernel. This allows that the ordering of the GEOM or ISD2 functions in the spectrum corresponds to the ordering of their numerical indices. This in effect may be thought of as a parameterization which specifies an ordering by its permutation indices.

This is analogous to weighting components of the combination function in accord with their magnitude.

It is conceivable that a function may be constructed for representing some experimentally determined points for plausibility judgements on specific examples (supplied by an expert). Given examples, an expert works out the confidences at various stages of inference across logical operators in corresponding rules. This gives a set of points in the plausibility space which represent plausibility function values. These points may be used for an interpolation or an analytic curve fit.

Some functions can be more easily handled if some restriction is imposed on the parameters. In many cases there is a symmetry assumption on the arguments, but it is also nice to have weights. For instance, given the function,

$$f(c_A,c_B) = \alpha c_A + \beta c_B,$$

with the restriction that $\alpha+\beta = 1$, one can see that there is only a need for a single parameter, since

$$f(c_A,c_B) = \alpha c_A + (1-\alpha)c_B$$

but, what we do not have the restriction. In that case, we may really want 2 separate parameters α and β) to bias the weighting of the 2 terms (c_A and c_B).

3.3. Search Efficiency

Even if one can not characterize the function space of interest completely in terms of a single parameterization, we may be able to specify a partial ordering over the spectrum of points of interest. This will help in the search of possibilities since now not every candidate combination function needs to be tried, rather only the most indicative functions are tried first, then the regions of the function space are rejected on the basis of the partial order, and other areas are explored further. Figure 8 gives the reader an intuitive idea of how much the proposed functions vary and suggests something of orderings among different functions. Some partial orderings for functions recommended in Chapter 1 are:

$$\ldots \leq \text{ISD4} \leq \text{ISD3} \leq \text{ISD2} \leq \text{MAX/p}$$

Functional	1.0 , 1.0 , 1.0	0.5 , 0.5 , 0.5	0.01 , 0.01 , 0.01	0.8 , 0.5 , 0.3	0.3 , 0.5 , 0.8
AVG	1.0	0.5	0.1	0.533	0.533
MIN	1.0	0.5	0.1	0.3	0.3
MAX	1.0	0.5	0.1	0.8	0.8
PSUM	1.0	0.875	0.271	0.93	0.93
PROD	1.0	0.125	0.001	0.12	0.12
GEOM	1.0	0.5	0.1	0.557	0.436
GEOM'	1.0	0.5	0.1	0.493	0.493
RMS	1.0	0.5	0.1	0.572	0.572
RM4	1.0	0.5	0.1	0.632	0.632
RSR	0.333	0.167	0.033	0.152	0.152
NRSR	1.0	0.333	0.1	0.456	0.456
ISD2	1.0	0.5	0.1	0.6	0.475
ISD3	0.555	0.278	0.055	0.356	0.244
ISD4	0.375	0.188	0.038	0.25	0.156
IMIR	1.0	0.680	0.278	0.752	0.604

Figure 8. Values Comparison of Various Plausibility Functionals.

$$GEOM' \leq GEOM''' \leq GEOMGF \leq GEOMLF \leq MAX$$

$$PROD \leq MIN \leq NRSR \leq GEOM' \leq AVE \leq RMS \leq RMC \leq RMQ \leq RMP_p \leq MAX \leq PSUM$$

A wise knowledge engineer would adopt a strategy to minimize total cost reflected by the number of tests *and* the computation cost per test. A weight corresponding to the computation cost of the functional might be applied to counting its significance in total search cost. Let us say we have an ordering,

$$f_1 < f_2 < \ldots < f_n ,$$

and the relative costs of single computation of the function are represented by $C(f_i)$. If one runs the whole mass of data through each functional to gather statistics about its performance, one might think of the problem as picking points from a number line which has been subdivided into regions

corresponding to the weights of computation. The regions corresponding to the cost of function f_i is denoted R_i. The task is to select a region between points A and B which is the "magic" region and to minimise the cumulative length of tested regions. The "magic" region is determined by a cost functional comprised of some combination of statistics from the evaluation of the performance of the combinational functions (see Chapter 6). A region R_m is magic iff the cost functional at R_{m-1} and R_{m+1} exceeds the value for the region. If the distribution of these functionals in real world problems was known, the global strategy could be specified by testing all the enumerations of testing orders and computing an expectancy of computation time (region length) for each. The maximum of these would be a good global strategy. We could assume a uniform distributions of these functionals , but that is obviously erroneous. A better strategy is to divide the functionals into 2 or more groups, and search the least expensive groups first, then the more expensive groups. The search strategy within a group might be a binary, or a more advanced search (e.g., Fibonacci search) if the number of functionals is extremely large.

The Figure 9 gives the relative computation times (based on a VAX750) of various operations recommended in Chapter 1. For order dependent operations, worst case sort time is included. From the table in Figure 9, it seems that a good strategy for choicing optimal AND/OR schemes, applying the heuristic mentioned above, would be use min/max first, then prod/avg, etc. .

Note that next to the MIN/MAX functionals the ISD2 functional holds the next most promise of being efficient. Only a add and shift (divide by 2) are required for each cycle. This assumes that the confidences in the expressions to be evaluated have been previously sorted (taking account of any weight to be assigned to different components). If the confidences are all static, this is easily done once for a knowledge base and never need be done again, unless the rule base is to be modified. In this case, ISD2 may be cheaper than AVG . However, if the confidences are determined dynamically, say by a function, then the sort needs to be done on the fly and AVG would be the cheaper.

Commonly, rather lengthy expressions of the form

$$\Sigma\, \alpha_i C_i < r$$

Execution Time in CPU microseconds					
Functionals	n=2	n=3	n=5	n=10	n=20
AVG	950	1120	1480	23604	43400
MIN	50	150	350	950	2000
MAX	60	160	430	980	2000
PSUM	40	630	1780	4650	10170
PROD	60	280	1780	2030	4000
GEOM	60	11020	33950	88160	186330
GEOM'	11320	11550	12400	13420	15500
RMS	11280	12050	12630	14760	18500
RMP	11260	11960	13360	16700	22600
rSr	500	1100	2380	5520	11330
N*rSr	720	1300	2550	5750	11340
NPrSrP	920	2180	4780	11520	23330
ISD2	50	880	2560	6780	14170
ISDP	60	860	2560	6720	14330
GEOM"	10550	11130	11760	12730	15000
GEOM"'	50	600	7720	94650	199670

**Note : n = number of arguments to the functional .

Figure 9. Time Comparison of Various Plausibility Functionals.

are evaluated to weight several conditions and check against a threshold. Note this is a *combination* (not necessarily a sum, but assume it is) over a series of identical combination functions. The α_i have been somewhat nondescriptly, but often painstakingly, and once determined, rarely changed. Moreover, once they are determined, the exact values are not critical, since the value of the left hand side expression above is never utilized beyond comparison with the right hand side. Further, since C_i takes on discrete (finitely countable) values, any number of transformations can be made to the α_i's and τ to simplify computation, so long as the computation gives consistent results.

Note that, as seen in Figure 9, the ISD2 function is quite efficient since it involves only an add and a shift (divide by 2) operation for each term. If one orders the α_i's, and modify them as follows:

0. m=1
1. find the smallest of all the quantities $k = \alpha_{ij}$, $\forall i,j \mid \alpha_{ij} \neq 0$
2. if $\alpha_{ij}/k = 1$ then $\alpha_{ij}' = m$ and $\alpha_{ij} = 0$
3. $m = 2*m$
4. if $\exists i \mid \alpha_{ij} = 0$ then goto 1

the result is a more efficient computation based on bit mainpulations. This allows a bit in a word for each weight. If there are more weights than word length bits, an ISDP scheme can be applied efficiently on integer weights. An appropriate method of adjusting the weights is to multiply them all by a constant K and truncating such that they all become integers:

1. sort all α_{ij}
2. find K
3. ensure that if $\alpha_{ij} + \alpha_{ik} < \alpha_{il} + \alpha_{im}$
   then $\lfloor K\alpha_{ij} \rfloor + \lfloor K\alpha_{ik} \rfloor < \lfloor K\alpha_{il} \rfloor + \lfloor K\alpha_{im} \rfloor$, $\forall$(i,j,k,l,m)

The trick would be to find the smallest such K which is appropriate, but fortunately, this is usually not needed, and a simple generate and test algorithm suffices.

1. sort all α_{ij}
2. find $K = 1/\min(\lvert\alpha_{ij}-\alpha_{kl}\rvert))$
   ensuring that if $\alpha_{ij} < \alpha_{kl}$
   then $\lfloor K\alpha_{ij} < \lfloor K\alpha_{kl} \rfloor$, $\forall$(i,j,k,l)

An example of the use of this algorithm is demonstrated in Figure 10.

Thus, taking the precision of the required computation into account, one can often simplify the evaluation to shift and add operators, generally improving the time efficiency by a factor proportional to the number of terms involved. Moreover, considering the recurring nature of such evaluations (rules are executed over and over within an expert system session), the overall improvement in the system turns out to be at least quadratic.

Given the rules (r=0.80),

$0.73\ [x_1=2,3] + 0.56\ [x_3=4] + 0.3\ [x_2=1,2] \rightarrow G_1$
$0.86\ [x_1=1,4] + 0.73\ [x_3=4] \rightarrow G_2$
$0.95\ [x_2=3,4] + 0.52\ [x_3=1,2,3] + 0.2\ [x_4=1] \rightarrow G_3$

we sort the weights,

(α_{34}=.2, α_{13}=.3, α_{33}=.52, α_{12}=.56, α_{11}=.73, α_{23}=.73, α_{11}=.86, α_{22}=.95),

and we choose K=25,

(α_{34}=5, α_{13}=7, α_{33}=13, α_{12}=14, α_{11}=18, α_{23}=18, α_{11}=21, α_{22}=23) ,

giving new rules (r=20),

$18\ [x_1=2,3] + 14\ [x_3=4] + 7\ [x_2=1,2] \rightarrow G_1$
$21\ [x_1=1,4] + 18\ [x_3=4] \rightarrow G_2$
$23\ [x_2=3,4] + 13\ [x_3=1,2,3] + 5\ [x_4=1] \rightarrow G_3$.

Figure 10. Example of the Use of Algorithm for Computation Efficiency.

3.4. Enhancing Decision Rules

Many additional features can be added to decision rules to further their utility in a real world system. These include weights at various levels of logical nesting and properties to control evaluation of rules. Some of these will be discussed herein. Further, the formulation of a conceptual model of an expert system can be quite complicated, and some of these concerns will be introduced as well.

3.4.1. Weighting Conditions

One can conceive of the partioning of the universe or event space at several levels, the selector level being the lowest level and complete event specifications (vector of values) being necessary to refer to an event. The highest level of partioning depends very intimately upon the application at hand and

misconceptualisation of the relevant subspaces is equivalent to violating the knowledge granularity structure of the problem [Michi 80]. The targets of the search of the space may be defined in terms of elementary conditions of the most basic attributes, or an elegant or elaborate system of weighted, multitiered rules may come into play. Often, it is very tricky to establish exactly the right context in which an elementary condition has relevance since the condition's relevance may vary with respect to the context involved.

One of the clearest models is a probabilistic one. The most elementary case is a uniform distribution over a set of values of a collection of variables and a uniform distribution over all points in the event space. Since a simple event in the event space can be represented by a selector, let us begin at the selector level and try to work our way up from there. The cross product of subsets of the value sets of the variables represents a set of conjunctions of selectors which together cover the event space. From these selectors, we can form a rule for recognising members of a subspace of the universe. For example, if the variable X has value set {a,b,c}, Y has {1,2,3}, and Z has {true,false} then the event space is $X \times Y \times Z$ = {(a,1,true),(a,1,false), (b,1,true),...} and the selector [X=a,b][Y=1][X=true] partitions the space by designating {(a,1,true),(b,1,true)} apart from the other 18 events in the event space. Figure 11 depicts a possible partitioning of a hypothetical events space and illustrates some problems which are detected only after the testing process is completed.

In the simplest case, the unit confidences (probabilities) are given by

$$p_i = m_i / n_i ,$$

where m_i is the number of values a selector might be satisfied with from the value set of variable v_i which has n_i values. Probability of a single compound conjunctive event is then

$$\Pi\, m_i / n_i$$

where i varies over all variables indexed in the selectors of the conjunction. The probability of all variables not explicitly mentioned in the term being satisfied is 1 as they are satisfied by any value. This gives a raw probability of actually seeing certain events which can be attributed to the rule and might be

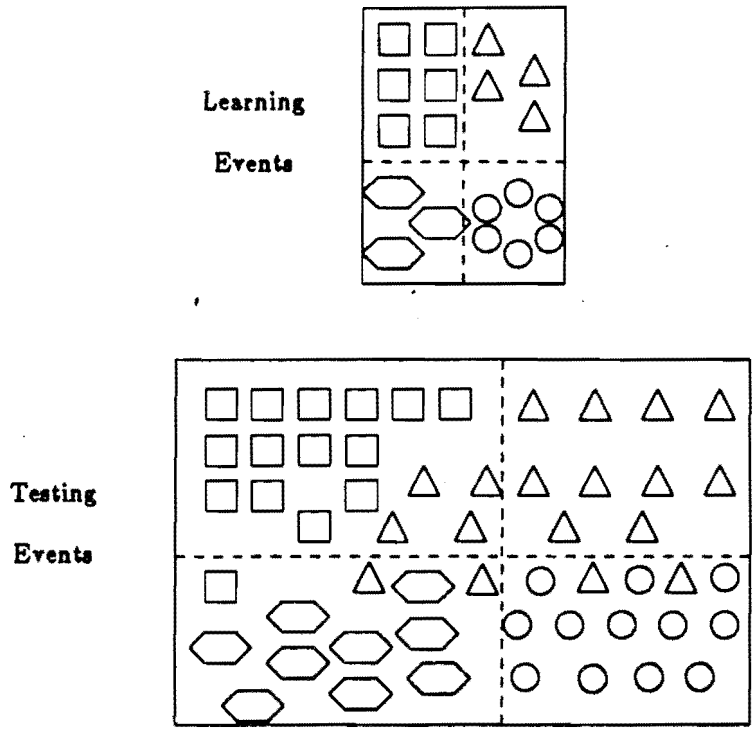


Figure 11. Partitions of Event Space.

combined with the confidence that the rule is actually *satisfied* if there is a need to distinguish two cases of equal (or near equal confidence). Similarly, these probabilities can be the basis of weighting the various components of rules, but rather in an inverse proportionality. I.e., those events of lowest probability should carry the most weight since they indicate high discriminatory potential.

Now consider a slight elaboration on this scheme. Suppose that in some absolute sense, the variables values are biased. That is, over all events, the variables takes on certain values more often than others. A distribution of probabilities gives the a priori probabilities of a variable, and might easily be handled above using

$$p_i = (p_{ij}) / n_i$$

wherever m_i/n_i occurs. It should be pointed out however that the notion of an absolute distribution over all variables is rather an unlikely one, since the interdependencies among various variables (or attributes) determines the probability of one given the others.

The contexts and distributions among the variables involved can almost always be specified by means of a set of background knowledge rules. These rules specify default values for one variable given another variable's value or specify which values of one variable preclude values of another. Alternately, they might specify the probability (distribution) of one variable's values with respect to other variables' values (often tabular in nature).

An example of a restriction rule as described above for plant pathology often arises:

> A leaf can have yellow spots, black spots, green spots or no spots. If any spots are present, then the binary descriptor, *leaf condition*, must have the value, *abnormal*. Yet, if spots are not present, then the same descriptor can be either *abnormal* or *normal*. Other descriptors such as *leaf malformation*, *defoliation*, etc. operate in a similar fashion.

Here, *leaf spots* is an attribute with 4 values, *leaf condition* has 2 values; yet, the total event space here is only 5 and the probability distribution (to be uniform) should only have 1/5 weighting to each of the leaf spot conditions.

3.4.2. Rule Base Propagation of Plausibility

The basis for most rule based expert systems is the inference rules of modus ponens and modus tollens. The fulfillment of the syllogism

$$\frac{A \qquad A \rightarrow B}{B}$$

is effected by obtaining A from input, having the implication as a rule in the system, deriving B from the control execution of the system. Uncertainty as to the correctness of condition A (based on inputs and

system state variables) or the validity of the implication (rule) intuitively inspires some uncertainty for the inference that assertion B (an action, decision, variable instantiation) is justified. Hence, the notion of a decision rule in the context of this paper includes an extended reasoning schema for propagating uncertainties accumulated either from surface data, implicit data (background knowledge & uncertainties), and weights and strengths of implication inherent in the rules.

As suggested earlier, uncertainty arise from imprecision, incompleteness, unreliability, perceptional or measurement aberrations, human error, and perhaps other sources. Imprecision ought to be avoided, but let us consider the others which one might label reliability or credibility or confidence. We can say that we have a degree of belief that arises at several different stages:

1) input – in the form of reliability, precision, or accuracy of source,
2) rule satisfaction – as previously defined,
3) rule strengths – as when one has more reliable rules and less reliable rules in the same set,

and in a sense, the propagation of the effects of one source proceeds sequentially, with the exception that (3) will cycle back on itself in a multitiered rule system. However, a good maxim that often applies is if all things are equal, no provision should be made to take account of them.

Imprecision comes from a conflict between (1) and (2). When we can only approximately assert A, we can not completely satisfy B. Symbolicly we might write

$$\frac{\simeq A \qquad A \rightarrow B}{\simeq B} \quad \text{or} \quad \frac{\simeq A \qquad \simeq A \rightarrow B}{B} \quad \text{or} \quad \frac{\simeq A \qquad \simeq A \rightarrow \simeq B}{\simeq B}$$

and intuitively, this has an appeal. If we know that the sugar concentration for optimal fermentation is 30% by weight, and the optimal temperature is 62° F, we do not balk if the brew master tells us that a vat is kept at 32% sugar and constant temperature of 65° F. We presume that because the conditions are *near* optimal that they are quite sufficient for a high yield. Such measures are assuredly subjective,

but a consistent set of axioms can generally be applied uniformly to solicit some reasonable results.

Reliability, human perceptional aberrance, etc. can proceed from either (1) directly, or from (3) indirectly as a bulk phenomena. The user or a demon supervisor of the user may decide at input that the data being entered is tenable yet somehow uncertain so a confidence (probability) is specified to indicate the possibility of the value not being correct. This is direct and basic specification of uncertainty for a datum. Similarly, when the rules are being entered into the system, a knowledge engineer may note a few exceptions to a generalization that may arise from intrinsic flaws of the generalization or some humanly motivated inconsistency which is foreseen (eg., *dark green* as a leaf spot color may be mistaken for *brown*). The inference mechanism might have weights and confidences hung at almost every elemental stage of the logic to cover all the possibilities so the inference might be

$$\frac{\begin{array}{l} A\ ,\ \alpha \\ (A\ ,\ z) \rightarrow (B\ ,\ y(z)) \end{array}}{A\ ,\ y(\alpha)}\ ,$$

where A,α denotes a logic expression A and its certainty α.

Another topic is incomplete or sparse data. Incomplete data has only to do with rule satisfaction. If the user misplaces or losses data, he would still like a decision based upon what he does have. If the expression used in place of A was in fact rather complex, we would like to complete the inference if most of the terms in A were satisfied, especially if none of our other rules came close to being satisfied. That is

$$\frac{\begin{array}{l} A_1 \,\&\, A_2 \,\&\, A_3 \\ A_1 \,\&\, A_2 \,\&\, A_3 \,\&\, A_4 \rightarrow B \end{array}}{\simeq B}$$

seems to be a fairly pleasing inference given that no other rules apply. Here as in all of the above,

human reasoning is more completely represented then in strict logic based models which demand complete and flawless data.

3.4.3. Distributions, Fuzz Factors, and Structured Variables

A number of innovations to further address the problems of handling plausibilities in a rule based inference system are put forth in this paper. Although the manifestations of the need for these variegated enhancements arose at different times and from diverse sources, they do share some commonalities that warrant simultaneous consideration.

For almost any multivalued variable, it makes sense to talk about a probability distribution for that variable, even a Boolean variable. For a 2 valued variable, we can imagine tossing a coin and counting the number of heads and tails and having this correspond to the relative frequency of occurrence of values of the variables. If a Boolean variable has a higher probability (expectancy) of one value than another, we say that it is *biased*. The generalization of an unbiased variable to a multivalued variable is a *uniform distribution*, ie. all values are equiprobable. And for any biased variable, there is some distribution that specifies what the probability of a particular value is. For continuous variables, there are distribution functions defined over ranges of values.

For variables with many values, the satisfaction of a selector can not be computed without respect for the variable's distribution in context. The deep structure of the model employed will provide some clues as to the significance of a context and whether the consideration is necessary. A three phase model of information transmission to the inner consciousness is depicted in Figure 12. The rightmost boundary is the barrier between what exists and what can be observed. This is partly based on the inferential powers of the human mind and past experience (including all abstracted experience, eg. linguistic). The middle boundary is the due to the discrepancy between what is seen and what may actually be there. This is due to aberrations in human perception (eg. optical illusions). The leftmost boundary is a product of human psychology. Due to one reason or another, a boundary exists between what we register by our

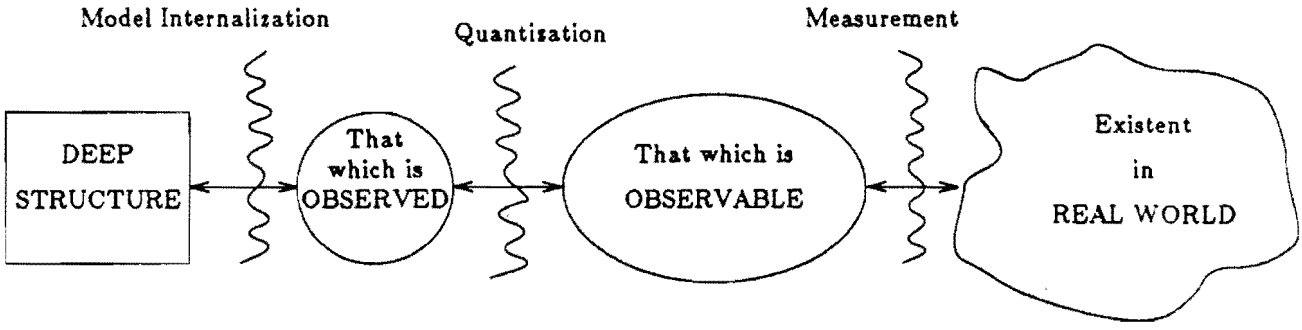


Figure 12. A Three Phase Model of Reality.

senses and what we accept in our minds. This is not always a paranormal phenomenon but can be rather subtle. The construction of a surface model concerns itself only with that which is observed and that which is observable. A crude manipulation of weights and strengths of implication based on statistical averages and summary statistics is very much allowed. Attempts to reconcile a deep model with *reality* must inevitably address all of the above entities and boundaries. Thus, the resolution of data and its internal format may vary, and attempts to satisfy all needs and uses of the data might fail under a single formalism.

Consider the variable TIME_OF_OCCURRENCE in the PLANT paradigm. It is a basicly tied to a more basic concept: Earlier vs. Later in Season. Giving the values for TIME_OF_OCCURRENCE as (MAY, JUNE, JULY, AUGUST, SEPTEMBER, OCTOBER) provides more flexibility than an exact date and removes a value judgement from a user (ie., whether the occurrence is early or late season). Since August 10 is still generally midseason and August 25 or later is probably late season, a question arises over exactly where the boundary lies in the surface model by the value AUGUST. This is easily resolved with a boundary-distribution function that maps the satisfaction of selectors at boundary conditions so

as to allow partial satisfaction of appropriate values. Rather than assigning units of plausibility at the logical condition level, this assigns it deeper still than at the basic selector level, since here it is associated with values of variables.

In some cases, a set of selectors may be so intricately intertwined in their interpretation (i.e., they form a context which can stand alone). Then, a multivariate distribution may be useful. The suggested default distribution is that of the multivariate standard normal. The means of this distribution would correspond to the centers of the interval of the reference sets of the selectors. The variance of each component would come from similar computations to the ones above, and normalization would be effected simultaneously with respect to all variables.

An alternative strategy to accommodate real valued (continuous) variables is that adopted in some systems called a *fuzz factor*. This feature defines the bounds of tolerance for any comparison test (equality, greater, less). In essence every selector, $[X \# x]$, is modified to be $[X \# x{-}\delta..x{+}\delta]$ where $\#$ denotes a relation and δ is the "fuzz factor". This fuzz factor can operate independently of the matching distributions or it can be intimately linked to the quantization Δ above.

The fuzz factor can operate independently in selectors of the type:

$$[X = x] \ , \ [X \neq x]$$

especially for real values and can amount to a rather strict matching distribution operator. It is unclear however if both fuzz and the standard matching distribution should be in effect during evaluation of selectors of the form:

$$[X > x] \ , \ [X \geq x] \ .$$

Structured variables can carry information about distributions and the values of the variable that should match other values. Since structured variables often carry partial information about the distribution of their values, they should be included in this discussion of the distributions of variables and matching distribution functions. Indeed, structured variables can have very odd distributions across values. Ie., the distributions may have the property that when integrated, they give a value greater than

zero. This is because sometimes structured values imply a greater or lesser degree of knowledge about the particular datum. Also, values at a given level may not be mutually exclusive, but set valued.

Consider the variables that might occur in the diagnosis of high voltage fault given in Figure 13. Various degrees of confidences arise from different reliabilities of various components (ultraviolet detectors, ozone sensors, current sensors, corona detectors) and various options and response times (alert operator, initiate auto diagnosis, slow power down, crowbar shunt) may be predicated upon the confidence of imminent failure (sustained pluming or arcing) and the cost of expensive equipment that is often at stake.

One distribution matching function spans horizontally across the values in the tree (Figure 13) as previously mentioned, but another matching function spans vertically in the tree. Here each underlying value of the variable implies all above values. A distribution function for the mapping might take into account the reliability of the device used to detect the various conditions or the human uncertainty (due to partial memory loss, or distractions at the time of observation) as to the specificity of a symptom. In this case, a uniform (50–50) distribution is quite tolerable at any node in the hierarchy as knowing that corona is present gives a 50% chance of either having a sustained arc or not, knowing that a sustained

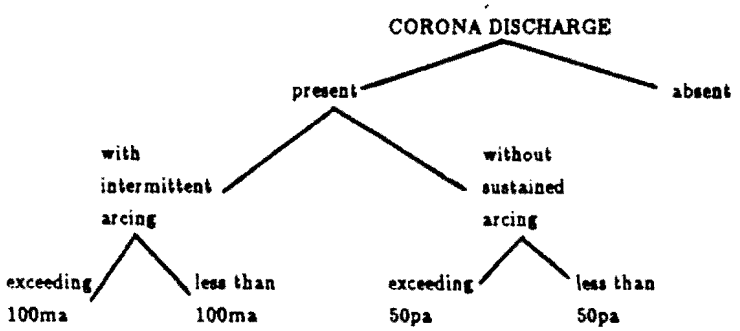


Figure 13. A Hierarchical Structuring of Variable Values Pertinent to Corona Discharge.

arc is present or not gives a 50% chance of the average charge emission being above or below characteristic levels.

3.5. Event Space Structuring

Consider the event space for a hypothetical example depicted in Figure 14. The objective is to learn from a set of examples and to test derived rules with a prescribed set of test cases. Regardless of the best foresight and attempts to be thorough and complete, an ideal structuring of the space is not likely, and

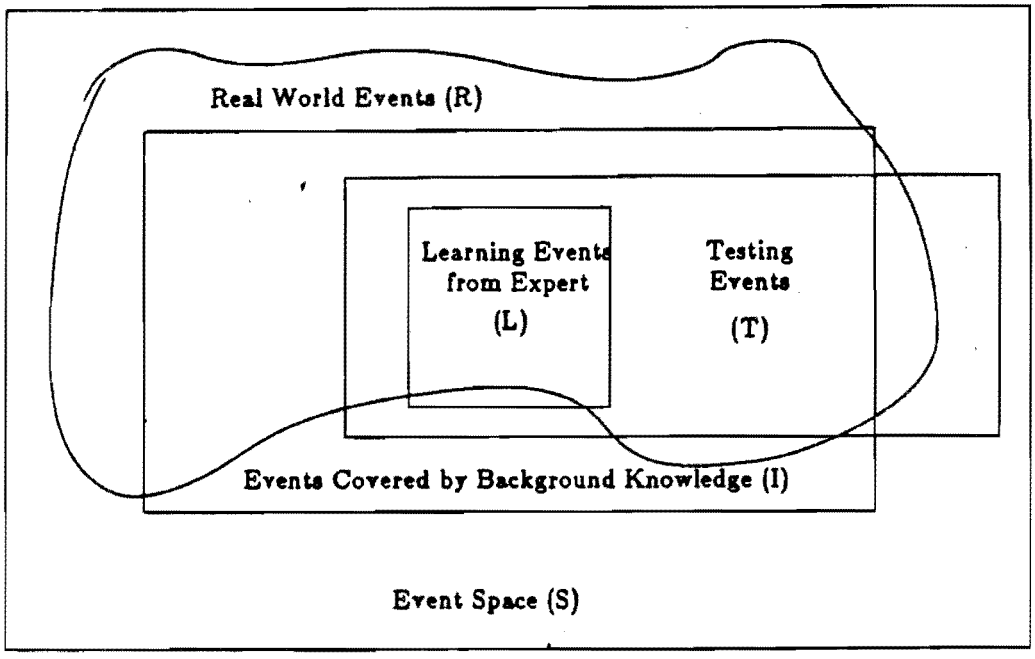


Figure 14. Hypothetical Event Space.

this hypothetical example is rather classic and paradigmatic. The various spaces overlap in various odd fashions other than the ideal which one would set in the mind. Ideally, the learning set is a proper subset of the test events which is a proper subset of *real world events* which is a proper subset of the *event space*. Here, the event space is simply the cross product of all values of the attributes, and the real world events are that set minus any disallowed combinations (for example, attribute *leaf spot* = *absent* and attribute *leaf spot color* = *brown* are not allowed simultaneously). The real world events are usually prescribed by a set of *background knowledge* rules which attempt to assign values which are compatible.

It is very common for experts to be rather inept at the process of formalising their own expertise [Michi 80], and it is a common occurrence to find that these event spaces are ill-structured. The background knowledge may be incomplete, and the learning events or test events may fall outside of what might actually occur in the real world. Usually, these problems work themselves out in the course of hashing through the data sets and rules several times. This is an inevitable fact of expert system development, and no way to automate this process will be proposed. Note that:

$$T \cup I \subseteq S \qquad L \subset T \subseteq S \qquad R \subset S$$

but

$$R \not\subseteq T \quad T \not\subseteq R \quad L \not\subseteq R \quad R \not\subseteq L \; I \not\subseteq R \quad R \not\subseteq I \quad T \not\subseteq R \; .$$

More will be said on this point later as it comes to bear on specific problems in plausibility representation and manipulation.

CHAPTER 5

A SIMULATION MODEL

5.1. Motivation

Given the success of a rule set and its corresponding evaluation function, one still wonders about the stability properties of these rules and plausibility functions in the light of changing data and new knowledge. For instance, suppose a domain expert realizes some little piece of context information which will make an existing condition in a rule much stronger in discriminatory power. Naturally, we want to incorporate this new piece of data into the rule. Since the plausibility functions were presumably derived from some global considerations of the rules and event space in the hope of optimizing a figure of merit, the new piece of the rule might possibly revise thinking on the matter. Admittedly, a certain number of these revisions may be made without any ill effects, but the process is somewhat like trying to build a house by simply nailing one board to another without any consideration of a global plan.

A similar situation arises for inductively (machine) derived rules. A new datum which is contradictory might enter the system in an online environment, or a particularly curious event is deemed worthy of being a learning event, but we do not want to take the trouble to perform the costly operation of rerunning the induction program. We could make an incremental change to the rule set, but this would again not take into account the global consideration of our existing plausibility functions.

Clearly, what is desired is some sort of estimation of the incremental effect upon the whole. This new figure of merit will aim at the approximate modification of the plausibility function or at least give a measure of the current utility of restructuring the plausibility generation algorithm. After noting a pattern of minor changes to rules causing gross jumps in the sensitivity levels, it might also indicate a need for reconstructing a set of induction rules, or the need to revise or restructure expert derived rules – but this is at the discretion of the knowledge engineer.

4.2. Analytic Techniques

The notion of the simplex tableau used in linear programming problem comes to mind as the sort of mechanical device that is desired. The tableau not only incorporates a method of solution and holds the answers to the problem in the last step, but also contains sensitivity information. The so called "shadow costs" or "marginal costs" indicate the reaction of the solution to slight perturbations of the input variables.

Two figures of merit are generally applied to rules in the abstract: a rule base is *consistent* if no two rules in the rule base conflict; a rule base is *complete* if there is no event which will generate a leading confidence below threshold. Figures 15 and 16 illustrate the evaluation of consistency and completeness for the simple case of unweighted, unstructured rules, having no plausibility propagation. Reinke [Re 84] gives additional consideration to the cases where the rules are structured (in which case the complementation operation employed in the completeness evaluation is not with respect to the entire event space but only subspaces). The real problems come with various plausibility schemes and weighting conditions (especially those where there is a functional relation governed by the variables

---

Given the rules ,

$[x_1=2,3] + [x_2=4] + [x_3=1,2] \rightarrow G_1$
$[x_1=1.,4] + [x_2=4] \rightarrow G_2$
$[x_3=3,4] + [x_3=1,2,3] + [x_4=1] \rightarrow G_3$

we generate the intersection of value sets of each selector, assuming a FALSE value for null sets arising from non-null selectors, and TRUE for null sets arising from null selectors,

$[x_1=3,4] + [x_2=4] + [x_3=2] + [x_4=1]$

is the complex indicating a consistency problem.

Figure 15. Consistency Evaluations.

Given the rules ,

$[x_1=2,3] + [x_2=4] + [x_3=1,2] \rightarrow G_1$
$[x_1=1.,4] + [x_2=4] \rightarrow G_2$
$[x_3=3,4] + [x_3=1,2,3] + [x_4=1] \rightarrow G_3$

we generate the union of value sets of each selector, assuming a TRUE value for null sets arising from non-null selectors, and FALSE for null sets arising from null selectors, and the result is complemented (assume values 1..5 for variables),

$[x_1=5][x_2=1,2,5][x_3=4,5][x_4=2,3,4,5]$

represents the events indicating a completeness problem.

Figure 16. Completeness Evaluations.

occurring in the conditions). Consider the two rules,

$[x_1=5:f1(x_2)][x_2=1:f2(x_1)] + [x_3=3:0.2,4:0.8,5:1.0][x_4=2,3,4,5] \rightarrow G1$

$[x_1=4:f3(x_2)][x_2=2:f2(x_1)] + [x_3=1:0.2,2:0.8,3:1.0][x_4=1,2] \rightarrow G2$.

Here the intersection and union operations are not so easily executed. In effect, we must perform the convolution of the functions f1, f2 and f3 over the value sets of attributes occurring in these rules in order to evaluate the overlap and subsequently apply the inverse plausibility functions to solve the equations of the unknown events giving rise to inconsistency or incompleteness.

If there are m variables, each with n values, the universe of interest is of the order n^m. Considering the operational difficulties of combinatorial explosion and the need to repeat the solution of a complex set of equations defining the optimal overlap, the systematic solution of this problem is equivalent to a nonlinear programming problem. Consequently, analytical techniques were set aside in favor of a simulation approach.

4.3. An Empirical Approach (Simulation)

A main result of experimentation is the ability to *bend* the rules (i.e., redefine the evaluation scheme) so as to minimise the number of clashes (2 rules simultaneously claiming the same event) while maintaining the maximum number of correct decisions (Figure 14 illustrates this notion). If a cost tradeoff can be arrived at, one can flag an irreconcilable event and throw it out. There is the additional desire to throw away as few events as possible. Actually, it is not necessary to throw away the event, but merely save the event for the next rule construction phase with remarks as to its aberration in the current test data. If incremental rule generation is available, it is used immediately, otherwise a file is constructed. The general philosophy is that aberrant test events should migrate back into the learning events or be summarised and displayed to a domain expert for his analysis and edification.

Another idea is to try to simulate the environment of the actual use of the rules and their associated control scheme. Incomplete data is easy enough to come by, simply take complete data and prune it back. Even plausible data is easily come by, one simply uses the background knowledge rules that are coded from an expert. For the incomplete events, the expert will eventually confess the cutoff point of his certainty as one subtracts information. But he does need to be prompted as it is very difficult for him to state right off just how much data must be removed until he is uncertain. The same process that is looking for misclassifications and clashes can certainly look for the events where the most clashes arise as the number of components of that event are gradually suspended.

A method for the above is thus proposed. Plausible events are fed to a chopper which cuts out random components of events by setting them to generic *UNKNOWN* values that are treated specially by evaluation schemes. The source of the events can be the learning events, test events from the real world (derived from online examples), or the plausible events alluded to above. This is motivated by a desire to further simulate the performance of the rules in the environment of a human user. Actually, the psychological models to be employed to this purpose can be somewhat more complex. Various thresholds and probabilities, as well as some more variables to model stochastic human mental processes

(forgetfulness, human confusion) and background knowledge rules to model human processes in terms of these psychological random variables, can all be incorporated into the model to give a much more sophisticated model to evaluate the performance of a set of decision rules in a simulated *real* environment in which they will be used. All this supports Minski [Min 73] in his claim that intelligent systems can not be designed to *learn* unless they can take into account the ultimate use and goals of the knowledge (system, learning), and one must recognize here that the ultimate goal is not simply *classification* unrestricted, but rather classification restricted by human data gathering skills (i.e., limited by certain types of incomplete, unreliable data). And note further, that there is no suggestion that the incompleteness and unreliability of data is absolute and holding infinite degrees of freedom. We are speaking strictly about those aspects that arise from the nature of humanity which (so psychologists belief) is limited. Put otherwise, *human frailty (boundedness) is not boundless : it has its limitations too.* And these limitations can be at least grossly modeled in a worst case sense, or a best case sense if you are an optimist. And these models can play a significant role in the simulation model that is proposed. Even more noteworthy is this role if the learning paradigm integrates the process of rule formation and rule evaluation in a continuous process of the induction algorithm used.

This is the crux of the methodology. A suggested computer driven model of this idea is illustrated in Figure 22. If one makes some assumptions about the distribution of events across classes, and no contradictory evidence to violate these assumptions is encountered in the course of the simulation (ie., running the exerciser), then one can apply conventional statistics to come up with a confidence interval based upon the number of events in the sample space (the number of plausible events used by the exerciser in simulation).

4.3.1. Simulation Details

The purpose of the simulation is 3 fold:

1) Exploring a large space for which some properties are desired,

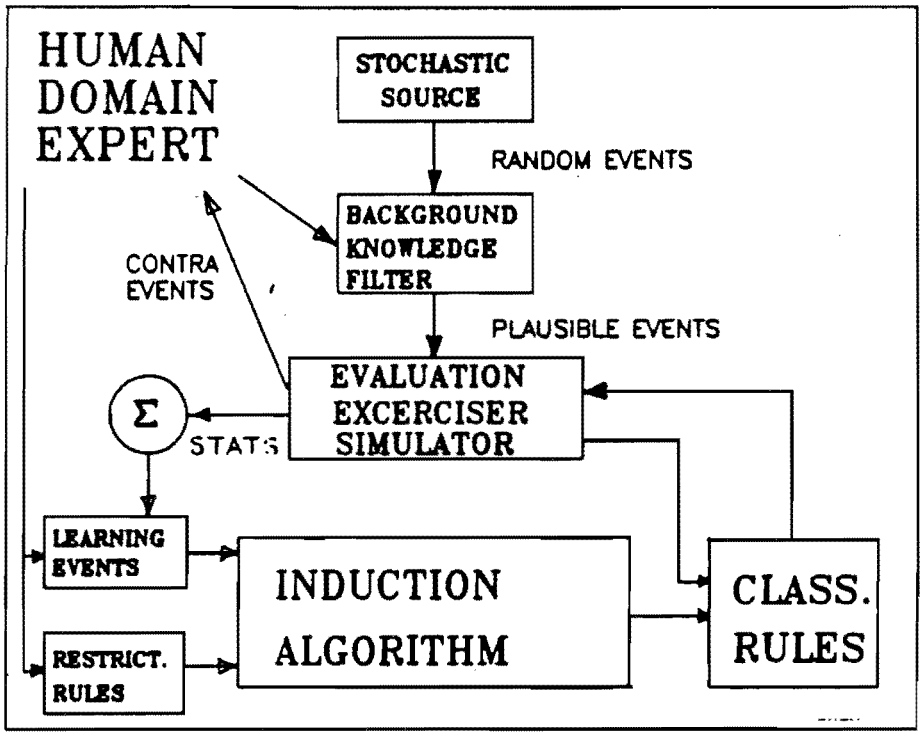


Figure 22. Computer Driven Model for Rule Refinement.

2) Semi–automated debugging,

3) Collection of exceptional conditions.

A stochastic process filtered by a set of restriction rules simulates the action of selecting test cases and verifying decision rules (e.g., for soybean disease decision). The decision rules are logic formulas that have been written by experts to recognise the patterns of abnormal conditions that specify the presence of a particular classification (disease) option. The elementary descriptors used in these rules are a set of (53) multivalued attributes which attempt to characterise decision events (diseased plants) in terms of observations that are easily collected (macroscopic inspection of plants in the field by an untrained

observer).

The descriptors and their values are presented in Appendix A for the soybean domain. The first step involves the coding of these values and the rules for classification (decision) in terms of a implementation language (PASCAL). My primary objective is to keep the rules close to humanly readable form. For the purpose of this translation, a parser was implemented to accept the rules and variable declarations in the format of the ADVISE [MiBa 84] metaexpert system. In fact, this is essentially Michalski's VL_1 formalism. The parser outputs Pascal code which is roughly equivalent to the original rules in structure, with the descriptors and values kept intact (i.e., not converted into numerics or otherwise symbolicly coded). The compiled code represents descriptor names and their values as integers. This allows existing test data formated for the learning algorithms to be used in the testing of the parsed rules. (E.g., AQ11[La 78] and CLUSTER[St 84] only accept integer valued attributes.) This representation also satisfies the needs of the simulation for a set of randomly generated values. No intermediate representation between variable values and values emitted by a pseudorandom number generator is required.

The mapping of descriptors and values to integers presents a few problems, but a few assumptions and restrictions resolve these. Consider the following variables and values.

Var_1	Var_2	Var_3
val_1	val_2	val_2
val_2	val_4	val_1
val_3	val_6	val_7
val_4		
val_5		

There is certainly no problem of assigning an integer to each variable,

$$Var_1 \rightarrow 1 \ , \ Var_1 \rightarrow 2 \ , \ Var_3 \rightarrow 3 \ , \ \ldots$$

but note the deletions of intermediate values for the ranges of Var_2 and Var_3. If the symbolic names of a variable are to be defined as constants to the compiler, one can not blindly associate the first symbolic

entry of a variable with 1, the next value with 2 , ... We must insure that there can be room in the interval of integer associated with a variables values to include all the values that may come later. The violation of order in Var_3 is estheticly disruptive, and it probably indicates an error in the thoughts of the originator. Thus, inconsistent orderings of this nature will be flagged as an error. Sparseness is perfectly easy to handle, one simply introduce a structure that represents the partial orderings that might exist. In fact, this is exactly the feature that a Pascal *set* has. Thus, one handles the deletion of values, val_1, val_3, val_5, from the suggested interval for Var_1. The poset structure constructed for each variable may be submitted to a topological sort and values assigned according to the resulting ordering. The association of a value with a number includes updating the set of possible values that a variable may take on. If values are associated with integers, this allows a more optimal use of storage since a minimum range of values is determined.

Details of evaluation code (both variable declarations and executable code) are included in Appendix B. As mentioned earlier, the decision rules rarely match exactly the conditions which they are intended to detect, yet they are formulated by an expert (eg., a pathologist) with the understanding that a good part of the rule being satisfied should be sufficient to suggest the classification (decision) pertinent to that rule. Thus, each rule upon execution returns a confidence value. Simultaneous recognition (non-disjoint classifiers) are allowed in the sense that more than one classification (disease) can be present (and often is), and further, interest is not only in the highest confidence possibility, but *all* possibilities with confidences above some threshold.

Although the rules used may be rather shallow, the evaluation functions used to define the partial matches are boundless. Further, more than a couple levels of logical operators may be involved, and the strategy for both parsing and rules execution should take into account these degrees of freedom to be left to an knowledge expert or knowledge engineer. Other assumptions attached to the logic system employed may profoundly affect the parser design and execution, such as the occurrence of a multiterm expression which employs a nonassociative operator. Still, one can generally match the allowed recursion

patterns of most logic to some form analysable by a stack machine, and furthermore each level of logic or each logical operator can be thought to have a stack associated with its evaluation. Thus, a single evaluation module for evaluating a logical level or operator need be written, and based upon the stack number or stack operator, the evaluation applies the correct functions over the stack contents. Atomic truths are evaluated at the lowest level, and combinant functions associated with AND, OR, IMPLICATION, etc., proceed from there. The user ought to be allowed to specify any of a number of interpretations for the generic AND, OR, etc. according to control parameters as suggested in the beginning of this paper. At the moment the allowed definitions of combinant AND and OR are sparse:

PSUM, MAX, MIN, AVERAGE, PRODUCT, ...

but these presently expand as experimental needs require.

4.3.2. Confusion Statistics

At this point, the concepts of consistency and completeness are hybridized into the notion of "confusion" in a rule base. By *confusion* , we mean the cases whenever the leading confidences of all options are of two or more decision options (e.g., diseases) that are within some tolerance δ. It is unimportant if there are 2 options within δ if neither of these has the highest confidence of all other options since it does not affect the ultimate decision. Furthermore, of the three following cases of confusion below, it is claimed that only case III is unacceptable.

Case I	Case II	Case III
D_1 0.10	D_1 0.98	D_1 0.45
D_2 0.08	D_2 0.95	D_2 0.44
D_3 0.03	D_3 0.85	D_3 0.10

Case I gives the leading options to all be below reasonable levels of confidence, so even though one is

higher than another, we can reject them all. This is an interesting scenario, and it indicates an incompleteness in the rule base, so these values must be printed out as they are uncovered in the course of running the simulation model (and delivered to the human expert to consider). Case II is unacceptable since it suggests that several diseases are strongly suggested by the symptoms that gave rise to these confidence values. This is not of particular interest on an event level (unless disjoint classes are to be enforced), but the frequency with which this phenomena occurs between various classes is a very useful sensitivity feature of a evaluation scheme for a rule set. Although the exact events are of no concern, the summary statistics, which is updated each time such an event is detected, capture this feature. Case III is the trickiest of the three. The confusion here is genuine and lies in the fact that more than one disease is suggested, but none of them clearly distinguishes itself above any others, nor is the confidence sufficiently high to warrant saying that it is any of them. Perhaps the particular events in this case are of interest, but clearly we want at least a summary statistic of the frequency at which this confusion occurs both between classes and on a whole. Thus, a matrix is quite appropriate in which all the cross confusion events are summarized for type II and type III confusion. If type I confusion occurs in a well developed rule base, it certainly does not arise very often, so individual events can be reported without overloading the human analyst. In new rule bases, the frequency of type I confusion will be quite high. It would be tedious to peruse volumes of exceptions, but a matrix summarizing the relative frequency of these cases against each other will be of little use. Rather, such cases should be compared against known cases of the classes and sorted according to a closeness of fit measure. Alternatively, a cluster algorithm might be invoked to find a conceptual clustering of the exception cases which might correspond to a class (disease) or a set of conceptually similar classes (classes). In this manner, the relative frequency of misjudgment can be surmised, and if desired, the exceptional cases can be viewed according to probable classification. This saves the expert some effort and helps him concentrate on the problem of rule revision rather than the analysis of masses of data.

4.3.3. Other Summary Statistics

Initially, a good definition of the ranges for the different confusion types listed above may be according to the scale:

type I 0.00 – 0.33

type II 0.67 – 1.00

type III 0.34 – 0.66 ,

given that little is known about the rules or their properties of sensitivity. The criteria for confusion is uniform across these types, so manipulating these thresholds does not serve to eliminate confusion, only to help the user think about underlying problems. Type II events are the most tolerable, and type I events are the least tolerable. In setting these thresholds, the user is fixing a value judgement for himself. If the user comes to the conclusion that type I events are irreconcilably dense (or in fact the union of any 2 types), then he ought to think about changing the rules radically, otherwise he might think of using a modified evaluation parameter and compare the results to see of the overall confusion is reduced to within reasonable bounds.

In addition to the statistics already mentioned for confusion, several other values have some merit in ascertaining the lea-way allowed in a rules set for various weights and implication strengths. I propose two modes for the process of collecting these statistics:

1) Stochastic Mode – embodied in the simulations approach already alluded to above wherein the events are essentially derived from a random source, some distribution functions, background knowledge and other human simulator elements,

2) Reference Mode – wherein the events are given by a human expert or other oracle (eg., another diagnostic program) and a "correct" or known decision is compared against decision of the rules.

The following summary statistics are felt to have some merits:

a) First Rank Hits and First Only Hits, b) Minimum Δ between leading decisions,

Average Δ between leading decisions,

Variance of Δ,

Maximum Δ

c) Minimum confidence among leading decisions,

Average confidence among leading decisions,

Variance of confidences,

Maximum confidence

d) Minimum threshold for clear decision

for stochastic events

e) Maximum threshold for complete decision

for test events

f) Average confidence for all events, all rules

g) Numbers of rejections (undiagnosable cases)

h) Expected number of rules involved in collisions

i) Expected rules to be involved in collisions

j) Expectation of collision for a given rule

k) Expectation of a decision option (i.e.,

# of decisions above threshold / # of events,

or

# of correct top decisions / # of events) .

Note that these statistics should be broken down according to each decision rule as well as amassed for the whole data set.

4.3.3.1. First Rank Hits and First Only Hits

The most interesting global measure of the performance of a rule set is the number of *first rank hits*. For control events, this is the fraction of all test events for which the correct rule evaluated to a leading hypothesis. For simulation events, it is impossible to define, however, of equal importance is the figure for *first only hits*. For control events, this is the fraction of all events which were assigned to the correct decision *and* were the only event in the leading confidence category (i.e., the leading confidence exceeded all other confidences by some specified value ϵ). In fact, a global measure of the product of these two metrics is quite useful in judging the relative improvement of one evaluation scheme of a rule set versus another.

4.3.3.2. Δ Between Leading Hypotheses

The difference of confidence values, Δ , for the two leading hypotheses serves as a reasonably useful statistic for debugging rules for which the user suspects false positives or confusion among the prime hypotheses. By *leading hypotheses*, let us denote those candidates with confidence greatest after sorting. Sometimes we may be interested in the top five leading hypotheses, but to remain consistent with the notion of type II confusion, one ought to count only those decisions with the top confidence greater than the threshold (>0.66). Over the leading decisions, four classical statistics are very conveniently computed and used: minimum, maximum, average, and variance.

For a given rule, a minimum Δ zero means that for one or more cases when this rule gives decision as the top decision, some other rule gives exactly the same confidence. Presumably, when this happens, the case is reported verbosely or as a case index number, but this figure tells much more when it is greater than 0. It then indicates the minimum separation of confidence values that has been noted between this rule and all other rules when this rule is the leading hypothesis. Conceptually, we desire this separation to be as large as possible, but based on the definition of type II confusion, a value of 0.33 or more would be ideal. In adjusting the rules, if we effect an increase in this value, we have produced a desirable change given that nothing else has been too badly damaged.

The maximum Δ is similarly of interest. A value greater than 0.33 indicates at least one case made the sort of separation of confidence that we would like. This is somewhat of a negative attribute since we are not interested in how high it gets once it gets above 0.33 . With values less than 0.33 , it shows that there are no cases where a disease clearly dominates the hypotheses. This represents the consistent presence of some sort of confusion which is undesirable.

The number of times the maximum and minimum occur are both useful statistics as well. This gives some idea of the typicality of their occurrence, but the user would really like to have some measure of distribution of values in between the maximum and minimum to prompt the decision to change the rules. In these cases, the average and variance give a better global overview to take into consideration.

The average Δ and variance of Δ help the user picture the frequency distribution of Δ. Of course, the interpretation of these statistics is up to the discretion of the user, but the same sort of concerns as above come to bear. Examples of a good distribution, a bad distribution, and a subjectively judgemental distribution are depicted in Figure 18. The area under the distribution curve which falls beyond the 0.33 mark may be used as a tolerable value to threshold against in resolving judgemental cases. For instance, if the area is less than 1/2 then the distribution might be plausibly labeled as an *operable* ("good") case.

The distribution itself is a debatable point. It might be Maxwellian, Poisson, or Normal as is convenient. But it should be noted that a small number formula for variance ought to be used. As pointed out earlier, the sample is generally a minuscule of the total event space. A binomial approximation of the normal distribution is probably tolerable for most applications.

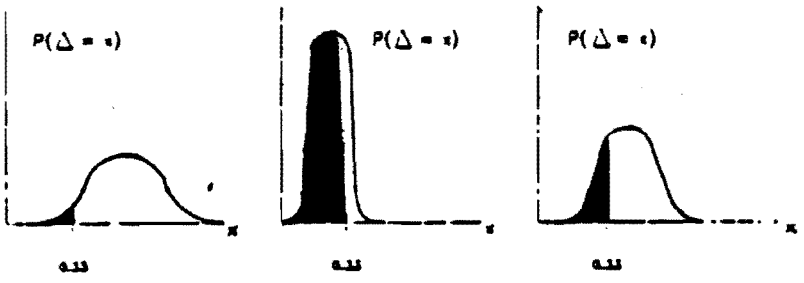


Figure 18. Comparison of Distributions.

4.3.3.3. Statistics of the Main Hypothesis

Another useful value to apply the min-max-ave-var analysis is the confidences of all leading hypothesis. It gives some measure of the possible effects of changing the threshold (0.67). The minimum confidence in the leading hypotheses indicates how close any rule comes to the threshold. If this value is exactly equal to that threshold, then raising threshold will surely cause rejection of some leading hypothesis for one or more cases. If the value of the minimum is above threshold, the threshold may be safely raised to this value. The maximum provides an upper limit on the threshold. If raised above the maximum, no case will be clearly diagnosable. A value of 1.0 for maximum means that at least one case is ideally diagnosable, a lesser value means all cases fall short of this ideal. Whereas the frequency of occurrence of minimum and maximum leading confidence, seems to be useful in judging the overall significance, the use of average and variance give a means of estimating the global distribution of confidences. Figure 19 compares the subjective case for Δ and overall effect of raising or lowering threshold. As threshold comes up we cut off some of the desirable decisions, but not to the same degree that the Δ values are enhanced. The weighting of such a tradeoff is also a subjective procedure. If the area B greatly exceeds A, then the savings may be significant, but the question is: "By how much ought B to exceed A to make a difference – or by what proportion ... ? " . The compromise also may exist across several decision options if a single threshold is used for all rules. In this case, perhaps the rules should be weighted in regard to their contribution to global statistics. What is better for one rule may be better for another and vice versa. The task is to find what is best overall. Alternately, this may be a case where the different statistics broken down on a per-rule basis are used to set different thresholds and other evaluation parameters for different rules. Global effects must still be taken into account, but per-rule figures can easily be combined later to amass the global figures at little additional computational cost. Local statistics should always be used first if possible, but in the cases of contingencies, global statistics serve to break ties.

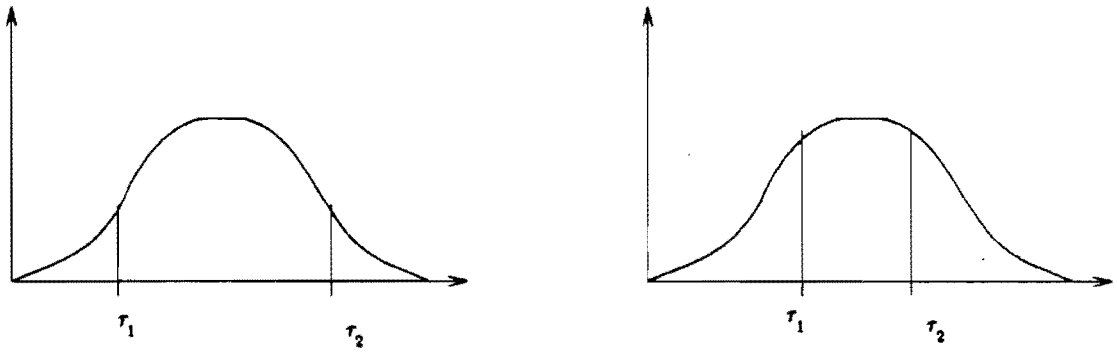


Figure 19. Effects of Thresholds on Distribution.

4.3.3.4. Threshold Statistics for Stochastic and Test Events

When stochastic events are being run and type III confusion is encountered, it may be the case that raising the threshold will eliminate the confusion. Thus, a useful statistic is the minimum of the leading confidences for type III confusion cases. Similarly, the maximum and average and variance can be applied, but the notion is less tractable than for the analysis of type II confusion already provided. The difficulty of trying to weight type II confusion summary statistics to compare against type III confusion statistics is again subjective.

The dual of this occurs when the user is running a set of test examples for which the correct results are known, a statistic to keep track of the maximum allowed threshold allowing accurate decision. This is simply the minimum confidence for all confidences of the correct decision rule over all cases.

4.3.3.5. Averaging over All Confidences

Consider the averaging of all confidences. The average confidence over all events should be quite small. Consider a typical case: If there are n rules and a confidence of 0.2 or less is expected for the n–1 rejected rules, and 0.9 or more for the true decision, then one would anticipate an average of

$$(0.2(n-1)+0.9)/n = 0.2 + 0.7/n \ .$$

The more rules available, the lower the average confidence. An average very much above or below this expected value would indicate a high confusion value. In fact, the confusion can come from either of type II or III, inflating the figure, or type I, deflating the figure. A low or high value does not necessarily indicate confusion, but it should be looked into. It can be indicative of very good results occasionally.

The distribution of values is also of value here, but simply recording the variance does not help. The distribution ought clearly to be bimodal, and it is wrong to assume too much about it. In fact, a true frequency distribution of the occurrence of various confidences is desired. But, a histogram will suffice and since there or 3 regions of interest, it would be best to make the resolution of confidence values be on the order of 1/10 or better. An ideal histogram is shown in Figure 20. Note the other distributions which are plausible. The mode at the low end of the spectrum indicates the cases where rules rejected the decision, and the high end mode represents the cases where rules returned high confidences (and presumably accepted the decision). Large differences in elevation between the low end and the high end represents possible confusion. A figure of merit to tract this confusion possibility is the integral from 0.33 to 0.66 (or other appropriate limits) normalized with respect to the sample size, alternately, the ratio of this integral divide by the sum of the integration from 0 to 0.33 and 0.66 to 1.0 . This gives an unbounded value that ranges from 0 to ∞, with 0 being ideal and 1.0 or less being acceptable.

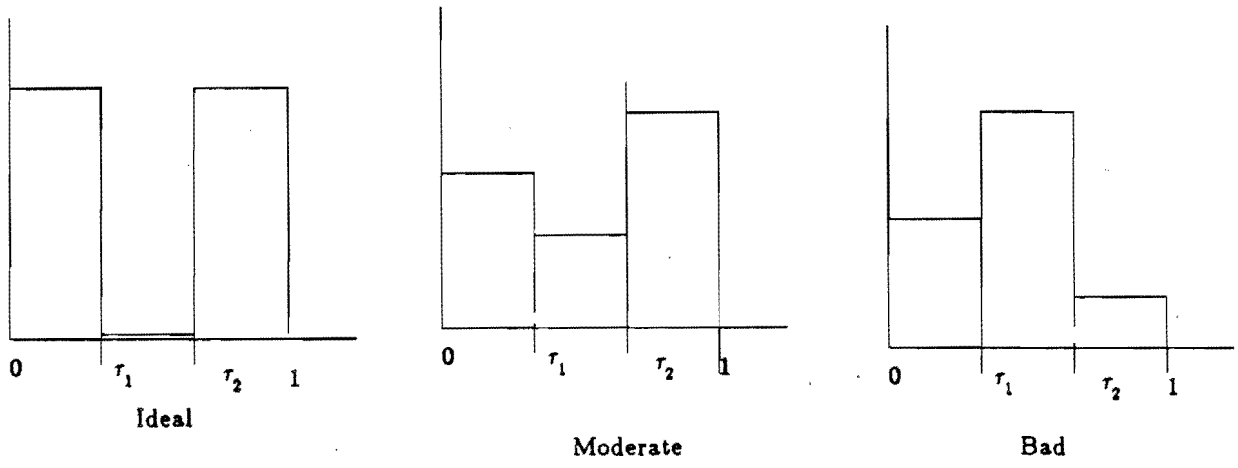


Figure 20. Comparison of Confusion Histograms.

4.3.3.6. Miscellaneous Counts and Expectancies

A few counts of various occurrences are of passing interest in lesser force than the above already mentioned, but often these numbers help interpret the above. The number of rejections help to decide if we are possibly overlooking something. If this number is very high, it may be due to the fact that a rule was produced in ignorance (supine or gross) of some categorical variant of a disease. This often occurs when we forget to take rare cases into account. Less often it may be due to an inexplicable or unique exception. If a previously unknown pathogen or vector is suddenly present, this too may produce symptoms previously inconceivable, giving rise to low confidence of all known rules. This is the response of humans to a disease like the AIDS outbreak in recent years.

Total number of collisions in each group gives a gross statistic, but if this is broken down by rule, and divided by total collisions or number of cases of the disease, it is more informative. For the latter, it gives an measure of likelyhood of collision for the disease. The former figure is like the conditional probability that any given collision will involve that disease. This represents the relative contributions of various rules to the total collision population. Relatively low values for a rule means a rule is not as involved in a collision as for some higher value. A sero value indicates a rule has not been detected in any collisions so far (and consequently, it is not expected to be in any collisions). Naturally, the ones with higher numbers ought to be attacked first or when considering the effect of perturbing a parameter by δ the maximum of these ought to have more weight than the minimum.

The total number of cases of a clear (i.e., unconfused) decision of a disease is simply the positive side of the same coin. After dividing through by total number of cases, we have a probability of clear decision for that rule. We ought to try to keep the rules with the high expectancy of decision in the forefront when considering changing various parameters. If we manage to increase this figure for a given rule, we have in a sense increased the stability of that rule, but we have done so at the expense of the stability of over rules. The ideal configuration with respect to this idea of stability is equiprobable occurrence of all rules.

4.3.4. LEFs and Goals

There are two approaches to using these statistics to evaluate performance for different values of evaluation scheme parameters. We might define a linear function which weights the different performance statistics or even a polynomial function. This function is then optimized via standard techniques over the space of evaluation parameters. But when the parameters are all being modified simultaneously, it is hard to resolve tradeoffs. This prompts a second approach which is multigoal oriented. A LEF (lexical evaluation function) can be defined by the user which specifies the relative significance of different goals. Where two goals conflict, the next lower goal in the LEF is evaluated. The values need only come within

some tolerance to prompt evaluation of the next lower goal. Thus, the goals are ordered, and a goal must be clearly dominate (exceeding tolerance) before it is the deciding factor. This assumes a linear (sequential) goal structure.

Often in robotics and other disciplines of automated planning, the linear goal criterion is too strict. The designation of *better goals* and *holding goals* is made to distinguish that a goal once established is either modifiable or not [Br 79]. The computation of the status of a goal is as follows for a linear goal structure:

1) set all goals $G_1, \ldots, G_n$ to be better goals ,

2) for i:= 1 to n satisfy goal G_i make G_i a holding goal.

The rather strict concept of a goal's transition from a better goal to a holding goal is easily suspended in favor of rewrite rules which specify preconditions on goals and specify when goals may be transformed for better goals to holding goals. This amounts to specifying the context under which a performance measure is the deciding factor. This is up to the individual user, but a few examples of criteria are those suggested in Figures 18, 19 and 20.

CHAPTER 5

RULE EXERCISER: SPECIFICATION AND DESIGN

5.1. Overview

Previous chapters expostulated various statistics or *figures of merit* for evaluating the runtime effectiveness of rules. Subsequently, a methodology for testing decision rules was put forth. It is the purpose of this chapter to describe simultaneously both the design and implementation of these ideas into a Rule Exerciser/Analysis Program (REAP) which will serve as a tool for the construction of effective knowledge bases.

This chapter describes the Rule Exercisor employed in the experiments discussed in the subsequent chapter. The motivation for the various features of the exerciser are presented along with the function and use of the feature. A more detailed specification of user input is detailed in Appendix B.

The Rule Exerciser allows interactive testing of rules under various evaluation schemes (see Chapter 1). Various performance statistics are collected and reported to the user. Based upon this information over the course of several runs, the user is able to select an evaluation scheme which has merits over others or the user can predict which rules need to be modified to improve performance. The last two chapters provide motivation for the use of test cases where correct decision class is known as well as stochasticly generated cases where the correct decision class is unknown. The primary purpose of the program is the latter, but provisions are made for the former, since often, when a user makes a change to the rules or evaluation scheme, there is a desire to check the performance against the known test cases.

The system is intended to be a user-friendly, self contained package separate from all other ISG packages for rule base maintenance. All of the users changes to rules and variables are made in text formated VL which is then translated into code for execution. All reporting of events is attempted at a

symbolic level except where the user has specified a terse format. Runs can be suspended, intermediate statistics generated, with the option to resume later. Various options for runtime testing of multiple rule sets and multiple evaluation schemes sequentially or in parallel are permitted. All of this is intended to make a flexible and versatile user interface.

5.2. Basic Function and Use of the Exerciser

The system must be fast and efficient. Virtually hundreds of thousands of cases are generated in a typical run, each one must be evaluated for every rule as well as tested for consistency with background knowledge. Large memory requirements are a tolerable sacrifice for improved speed, so verbose and even cumbersome code with little interpretive component is allowable. Ultimately, the Rule Exerciser is to be installed in the ADVISE system, so compatibility with a micro computer system is desired. In fact, the rule evaluation code is identical to that used by the IBMpc version of PLANT/ds, a subsystem of ADVISE (see Chapter 8). In fact, the parser is nearly identical as well for the two.

The exerciser has 2 basic modes:

1) Control Mode – which test rules over a set of exemplars for which the correct diagnoses are unequivocably known,

2) Simulation Mode – which tests the rules over a much larger set of stochastically generated *plausible events* for which the correct diagnosis is unknown.

The use of Control Mode is somewhat obvious. For a new set of rules, it provides a way of quickly gathering information about what ought to be modified in the rules to make them consistent with the known exemplar cases. Simulation Mode is a means of evaluating the completeness of the rules. If there exist cases where the rules will overlap and *confuse* the decision confidences or fail to provide significant confidences that the rules are clearly satisfied, then the simulation stands a good chance of finding such cases if they are numerous. Furthermore, the two modes together provide a tool for gathering comparative data in these respects. When a change is made either to the rules or to the evaluation

scheme, we compare results of Control Mode, before and after the change, to ensure that CONSISTENCY is maintained (or improved if possible), and in Simulation Mode, that COMPLETENESS is improved (i.e., that *confusion* has decreased overall). In both modes, one is generally concerned with GLOBAL EFFECTS throughout a set of events. In the case of Control Mode, one wants the total number of false positives and false negatives (or some functional combination of the two) to approach zero. In Simulation Mode, one wants the *combined confusion* (or sensativity, or specificity, etc.) to tend towards zero, contingent upon maintaining the same levels (or better) of CONSISTENCY in Control Mode. (Actually, here again some functional combination of the consistency figure and completeness figure monotonicly increasing may be acceptable.)

Despite these global or holistic concerns, human intervention is required to decide the significance of isolated, individual events. These events shall be designated as *exception events*. The reporting of these events can be either verbose, terse, or nil. The complete reporting of events includes all of the variables and their values translated from the numerical internal representation to the verbose English or symbolic designations with which the domain expert deals. An appropriate label or header is attached to these, and they are dumped to a file for later perusal by the expert. It would be too cumbersome for the user to deal with anymore than the most general statistics describing the number or distributions of confidences for such events. This extends to a general philosophy that a terse form of reporting is useful since the number of exceptions maybe large, and it is uncertain if all of the verbose exception output is of any use (particularly when one is first beginning the testing of a set of new rules and everything after the first few exception is essentially redundant). All but the most general of the mainline output will be directed to files for later viewing and analysis, avoiding the tedium of seeing undesired results, but preserving them for later reference if desired. Here, only the numerical representation of the events need be reported. At a later time, this numerically coded information can be decoded into the more verbose format. For extremely large exception sets, the terse form is much more convenient. For n-variables, each event is an n-digit number which corresponds to the current random number generator seed in simulation mode. Given that all events up until a certain event have been debugged or brought to bear

on the rules, the random number generator can be reloaded with this seed value, and the simulation can proceed with out reduplication (loss of CPU time). Furthermore, since the simulation can be restarted at an indefinite time in the future via this mechanism (i.e.. knowing the seed and intermediate results up until that seed), there is no problem with interrupting the process at any time and storing the state. This is crucial to long runs for which total runtime may be inestimable, and yet when the process has continued into prime-time computing, one wishs to terminate it and restart later at the *happy hour* rates.

Two possibilities occur for firing the rules while testing both in stochastic and control modes. In a *parallel* manner, each event to be tested can be tried in each rule in turn before going on to the next event. Alternately, in a *serial* manner, all the events can be tried in the first rule, then all the events tried in the second rule, and so on, until all events are tried in all rules. For the control mode, the serial fashion is used. This allows some results to be reported as soon possible. For the stochastic mode, the parallel fashion is used because the events to be used are far more numerous than in control mode (this is the general nature of simulation models). Consequently, there is no advantage to getting results as soon as possible (there is little advantage to getting results after 1 hour - it can not be interactive). Furthermore, there is an advantage to the parallel firing of rules since there is computation (calculation of random numbers) involved in sequencing through the events, and in serial fashion, the sequencing would have to occur for each rule.

5.3. Rule Parser

As suggested earlier, since interpretation of logic formulas and rules is very inefficient for this application where every rule is evaluated thousands of times, the VL rules (Chapter 2) require some translation before they can be used by the rule exerciser. In addition to the rules, a specification of the symbolic and numeric variables used by the rules accompanies the rules, and in some sense, this too must be translated into Pascal code.

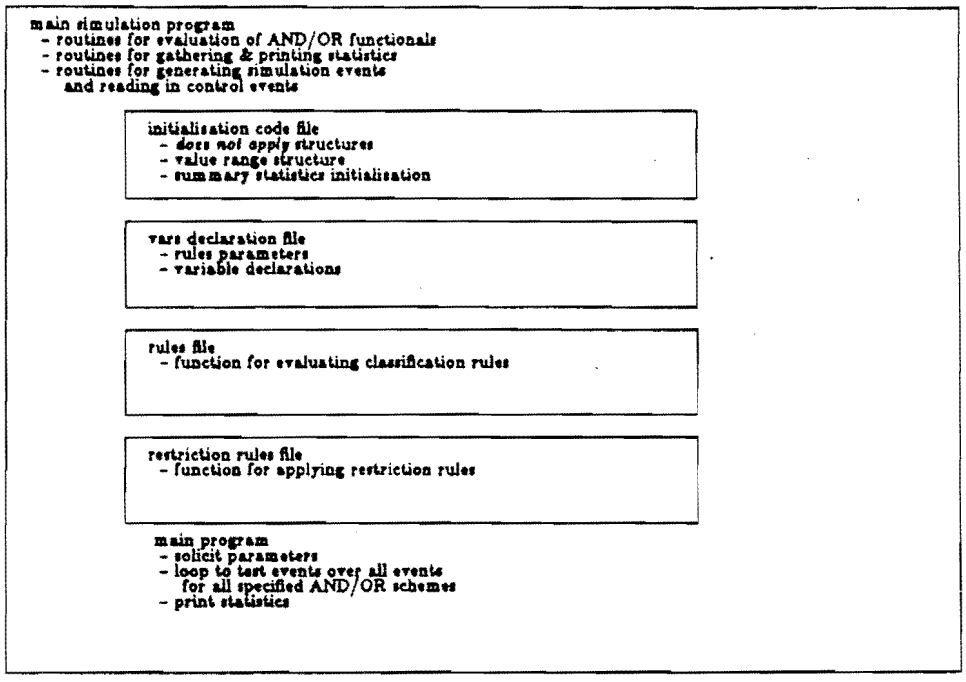


Figure 23. Overview of the Rule Evaluator Program Architecture.

A fairly flexible rule testing program depicted in Figure 23 was devised to exercise (test) the rules over both simulation events and control events (the learning events in this case). A parser was written with the aid of YACC to reduce the representation of rules as in Figures 24 and 25 into a Pascal function for evaluating a specified rule. This Pascal encoding of the rules was included in the main compilation of the rule evaluator program. The parser also generates code defining the attributes as Pascal records and code to initialize peripheral data structures. A provision is made to parse restriction rules as well. These are a useful means of providing background knowledge before executing a rule. The form of such a rule is typically :

$$[X > \text{average}] \rightarrow [Y = \text{does\_not\_apply}][Z = \text{does\_not\_apply}] \ .$$

This says that if the value of X is seen to be greater than the attribute value "average" (a particular

```
file → block | file block
block → VARS varsbody END | ID RULES rulebody END
rulebody → rule | rulebody ; rule
varsbody → vardefn | varsbody ; vardefn
rule → lms :> | ID = idnode | | ϵ
lms → lms V optreal condstmt | lmparts
lmparts → optreal condstmt | lmparts + optreal condstmt
optreal → REAL | ϵ
condstmt → condition | condstmt condition
condition → selector | ( term )
selector → | idnode rel ref optweight |
optweight → : REAL | ϵ
term → selectors V zterm | selectors => zterm
zterm → selectors | zterm V selectors
selectors → selector | selectors selector
rel → = | > | < | <> | >= | <=
ref → ID ddval | ref , ID ddval
ddval → .. idnode | ϵ
vardefn → idnode dfield optprops | ϵ
dfield → INTERVAL = intreal .. intreal | NOMINAL = ( idsunique )
optprops → props | ϵ
props → prop | props prop
prop → PROP = propid %%
propid → TRANS | PROMPT
idsunique → ID | idsunique ID
intreal → INTEGER | REAL
```

Figure 24. Grammar for Decision Rules.

value of the linear variable X), then set both the value of Y and the value of Z to "does_not_apply". Such rules are of great value as a means of handling irrelevant, unknown, or noisy values, in addition to the obvious use in representing background knowledge. Contrast the simple form of these restriction rules with those of decision rules in Figure 25.

The input to the rule evaluation includes the specification of the following items:

1) whether to throw away exception events or store them

2) whether to do testing against random or control events

(number of events for run or name of control events file)

3) whether to start from scratch or continue halted run

4) thresholds for the confusion statistics

5) enumeration of which AND/OR schemes to test.

```
0.6 |TIME_OF_OCCURRENCE < = OCTOBER | |STEM_CANKERS= ABOVE_SECOND_NODE|
    |CANKER_LESION_COLOR= BROWN|
    *
0.3 |PRECIPITATION < = ABOVE_NORMAL | |FRUITING_BODIES_ON_STEM= ABSENT|
    |FRUIT_PODS= NORMAL| |TEMPERATURE> = NORMAL|
    *
0.1 |CROPPING_HISTORY < = FOUR_OR_MORE| |PREMATURE_DEFOLIATION= ABSENT|
    >
    |SOYBEAN_DISEASE= DIAPORTHE_STEM_CANKER|;

0.6 (|TIME_OF_OCCURRENCE= MAY AUGUST| = >
      |TEMPERATURE> = NORMAL| |CANKER_LESION_COLOR= DARK_BROWN_OR_BLACK|)
    |PLANT_HEIGHT = ABNORMAL| |CONDITION_OF_LEAVES= ABNORMAL| |CONDITION_OF_STEM= ABNORMAL|
    (|TIME_OF_OCCURRENCE > JULY| = >
      |LOCATION_OF_EXTERNAL_STEM_DISCOLORATION= TILL_ABOVE_SECOND_NODE|)
    |CONDITION_OF_ROOTS= ROTTED| |PREMATURE_DEFOLIATION= ATTACKED_BUT_DEAD_LEAVES|
    |EXTERNAL_DISCOLORATION_OF_STEM = BROWN| |LEAF_WITHERING_AND_WILTING = PRESENT|
    *
0.1 (|CROPPING_HISTORY < = FOUR_OR_MORE| = > |SEVERITY = SEVERE|)
    >
    |SOYBEAN_DISEASE= PHYTOPHTHORA_ROT|;
```

Figure 25. Typical Rules.

Note that the capability to halt a run and resume it is provided. This is accomplished by signaling the program to save its state and flush all buffers. This is highly desirable if runs (particularly simulation runs) are expected to last on the order of hours. If the system crashes in the middle of a lengthy run, a lot of computation time may be lost unless the state of the run is periodically saved. The other point of interest is item 5. Currently, a vector of AND schemes and a vector of OR schemes are solicited as input. The program uses the cross product of the two schema vectors in the evaluation of rules.

A parser which creates several code files to be included in the compilation and load sets of the rule exerciser solves the problem of translation and efficient execution of the rules. The code produced falls into the general program flow as a series of Berkeley Pascal *include* statements, the structure of which is detailed in appendix B. The file vars.i declares the variables and their values as they appear in the rules. This file ought not change too often and is strictly a series of declaration statements. The rules.i file is suspected to change more frequently. It is meant to be separately compiled after the first time, and the resulting object file simply loaded with the rest. The only time that everything need be recompiled is when new variables are added to the rules or the allowed values are changed. Currently, all of this is

handled automaticly with a Unix shell script.

Actually, several other peripheral files exist in addition to the rules.i and vars.i files. The background knowledge rules assign values to other variables based upon some familiar condition which occurs in terms of values already known. These rules differ from the decision rules in that they are strictly deductive and not intended for inferential use. No confidence is associated with these rules because they are hard and fast. These rules are parsed in a different manner and the output is the file restrc.i . Additional attributes of variables are recognized and assigned via the file init.i . For instance, the value DOES NOT APPLY is sort of an odd value for a variable to be allowed to have. A flag is associated for this purpose in the declaration of each variable, but ordinarily it is disabled. If it is to be turned on, a statement to this effect occurs in the init.i file.

5.3.1. Processing of Variables

The evaluation mechanism is based upon the correspondence between the symbolic values of variables in the rules and the numerical values of the variables in effect at runtime. Using the algorithm proposed in the last chapter for the purpose of defining this mapping, an extension to the parser converts the declarations in the parser input to the "vars.i" file which is included in the constants declaration in the Pascal code.

For the sake of uniformity, a variable is represented as a record which contains the variable value along with other attributes of the variable. Variable are assigned integer values in a finite range, usually 1..10. The possible values of the variable are represented as a set. Conceivably, in the future, the ability to employ metarules to dynamically modify variables value ranges in appropriate contexts will be added. Each symbolic value is mapped to a numeric value on a per variable basis, and each variable is assigned a value which indexes into an array of variable records. For readability of the code, integer constants are generated for each variable and each value and these symbolic values are used in the rules. In order to accommodate an 8 character symbol restriction of many versions of Pascal, these symbols are

abbreviated in accord with a simple algorithm.

In addition to the conversion of symbolic values to numerics, other considerations exist in preparing the variables for use in the Pascal code for evaluating the confidences of rules. Whether or not a variable value allows one of the metavalues such as *DOES NOT APPLY* as an intrinsic value or not is an attribute that should be processed in scanning variable definitions. This property on a variable is represented in the dnaflag field of the variable's record. Initialization of such fields as well as the initialization of the summary statistics accumulators is contained in the file init.i.

5.3.2. Processing Restriction Rules

Currently, the processing of the restriction rules results in the construction of a file restric.i containing a procedure restrictionrules which takes an integer corresponding to a variable as its argument. This procedure is simply a series of if–then assignment blocks which are rather literally transcriptions of the restriction rules.

Although in the current implementation, the restriction rules are applied in turn to each variable for each event, there is an optimal order for applying restriction rules, and a set of conditions can be derived to effect the minimal number of restriction rule applications for a given state of variable values – even including unknown values. For instance, certain variables are more important than other variables. Certain restrictions may invalidate others. The more relevant variable's restriction rules should be applied first.

5.3.3. Processing Decision Rule

Parsing the rules proper results in a file rules.i . This file contains a function firerules which takes an integer corresponding to a decision variable value and returns a real number corresponding to the confidence that the rule is satisfied. The code for the rules contains symbolic values of the variables and can easily be compared to the original rules. This is handy if one desires to rather quickly adjust some

weights.

Rule evaluation is accomplished by pushing/popping AND/OR operands onto and off of stacks corresponding to various levels of depth in the logical expressions. The various schemes described earlier are embedded in functions which represent the various levels of AND/OR operators. These functions use the specified scheme for pulling values off the stack and combining them into values which the rule evaluation function uses to push back onto the stack or return as the final value for a rule. Theoretically, every level of AND/OR evaluation could be treated separately, but currently only 2 levels are supported for each. The use of set operators to compare the value of a variable with the value sets in the rules helps make low level evaluation relatively efficient.

5.4. Experimental Goals

The design presented reflects certain experimental goals, more than future utility of the program. The issue of particular interest is the prospect of running random, plausible events through the evaluator and seeing how much confusion there is in the rules, and altering the evaluation parameters along a spectrum to minimize a summary statistic. If this proves feasible, the idea of a spectrum can be generalized to a multidimensional space by considering a spectrum of functions for each evaluation level and/or each logical operator. The simulation run for each of these points represents a value of the confusion (perhaps multivalued) point in a multidimensional space. We are interested in minimizing this confusion, so one can effect a discrete gradient search to home in on at least a local optimal point (i.e., a specification for the evaluation scheme to be used for a rule set).

Thus, let X denote a specification of the following items:

1) any parameterisations of combinant logic evaluator functions,

2) designations of which functions are to be associated with a level,

3) any weights associated with a context or term or atom,

4) general treatment of unknown values (e.g., best/worse case analysis) .

The confusion vector, $\mathbf{C(X)}$, is then the result of running the simulation with the parameters defined by $\mathbf{X}$. Further, later a cost function $U(\mathbf{C(X)})$ which evaluates the utility of a particular confusion vector is defined. Thus, taking

$$\Delta = \alpha \bullet (U(\mathbf{C(X+\delta X)}) - U(\mathbf{C(X)})) \simeq \alpha \bullet \nabla U(\mathbf{C(X)})$$

and optimizing

$$U(\mathbf{C(X+\Delta)})$$

with respect to α, one can stepwise converge on the optimal $\mathbf{X}$ (parameter values) to minimize the overall effect of confusion.

The motivation for this is that when an expert derives the rules initially, he is only envisioning a small learned subset of the entire space (learning set). In fact, the rules produced by Michalski's AQ11 program hold the same limitation. The point is that little or no confusion exists in the learning set, but due to the means by by which these *characteristic* cases were chosen, a much greater degree of confusion is likely to exist over the universe of all possible events. It is conjectured that this confusion over the whole space may be eliminated, reduced, or at least whittled down by varying the above mentioned schemes of evaluation so as to maintain consistency with the learning set. Actually, a cost tradeoff between local, learning set consistency, and global confusion may be necessary in most real world problems. This is a psychological aspect that is unpleasant to most experts, but is probably inevitable.

As a matter of procedure, before any confusion statistics are given to an expert, some sort of variance reduction or averaging of results of multiple runs with different seed values should be performed. This should be predicated upon a 0.05 level for a χ^2 goodness of fit for multiple runs of the simulation. Various sample sizes differing across at least 3 orders of magnitude should be considered. The lowest order of magnitude should ensure that at least 3 events of any class are generated.

The utility function over the vector of values may have some strange behavior that contributes to an instability in the convergence or a divergence. For example if an a priori relation exists between 2 components, and they have equal weights in the contribution to the utility function, trying to follow

incremental changes may lead us in circles. Hopefully, at least the other components would converge if these unstable elements could be held constant. Fortunately, several variance reduction techniques may be applied to this purpose. One of these is the idea of a *lexicographic evaluation function* [MiRe 84] for the composite utility. The LEF consists of an ordered sequence of the elementary criteria along with tolerances which control to what extent nearly optimal solutions are considered equivalent.

CHAPTER 6

TWO EXAMPLES: SOYBEANS AND MONKEY

6.1. Overview

This chapter describes the analysis of two case studies in rule refinement, both conducive to the application of techniques discussed earlier. The first study is a rule base derived from automated induction over a mass of behavioral data for a small group monkeys in Lincoln Park Zoo. The other study is a rule base from the expert system PLANT/ds [Ch 79, ChMi 80, Uh 82, Re 83] which is a real system for assisting plant pathologists in the University of Illinois Agricultural Extension Service. The relatively sparse reification provided by these studies suggests the need for future consideration of these techniques in similar problems and even more complex problems.

6.2. Monkey Data Results

As a preliminary test of the the proposed methodology, an experiment which involved the use of machine derived (induction) rules was run. A set of learning events consisting of 1,815 observations of monkeys at the Lincoln Park Zoo in Chicago were used as input to the GEM inductive learning program [MiRe 84] developed at the University of Illinois. This data consisted of values for attributes recorded during the summer of 1983 characterizing the activities and environment of the animals at various times. The primary target for classification rules was taken to be the attribute "FOCAL ANIMAL". A complete list of 15 attributes are shown in Figure 26.

The events were partitioned according to the "FOCAL ANIMAL" attribute, and the "SPECIES" attribute was eliminated from the data (it is redundant). GEM was applied to obtain disjoint covers. The rules obtained are shown in Figure 27. Note that the rules contain relatively numerous terms. Since no domain expert was available (anthropologists volunteered the data with various disclaimers), no

Attribute	Number of Values
focal animal	4
species	2
hour	10
activity	42
participant	8
neighbor	8
distance	10
division of cage	10
support	10
general act	28
hyperact	7
temperature	10
humidity	9
precipitation	4

Figure 26. Attributes and Values for Monkey Data.

attempt was made to augment or qualify the rules. In a more rigorous series of experiments, a random partitioning of the data into a set of test events and learning events would be desirable (in fact such experiments are currently in progress), but all of the learning events were used as test events as well.

In the course of setting up experiments, several modifications to the proposed method were introduced. First, it was discovered that thresholds should probably be set according to the results of the control events. A phenomenon was observed whereby the performance of confusion matrix off-diagonal error was parameterized by threshold values. As the threshold went from 0 to 1.0 a steady improvement in performance was noted as expected, but then above a value q, performance remained constant with respect to further increase in threshold until a critical value was reached whereby the main diagonal performance rapidly degraded. The lower bound on this *quiescent* region (referred to as the q-threshold) was found to represent a significant threshold when later experiments were carried out with the simulation mode testing. as it is close to the optimal (quasioptimal) threshold for a given scheme and hence is a good starting point in further searches (for thresholds or whatever). With respect to

[day=0] & [hr=0..3] V [day=2..3] & [hr=2..9] V [day=0..7] & [hr=6..9] & [act=4..40] & [neigh=2..5] V [day=0..7] & [hr=6..9] & [div=5..8] & [sup=0..7] V [day=2..7] & [hr=6..9] & [act=0..16] & [neigh=2..10] & [dist=0] & [div=0..1] V [hr=6..9] & [neigh=0..2] & [genact=0..3] & [pcpt=1..2] V [day=0..7] & [hr=5..9] & [neigh=0] & [sup=0..5] V [day=0..7] & [hr=6..9] & [act=14..40] & [neigh=4..6] & [div=0..3] & [genact=0..4] V [day=2..7] & [hr=6..8] & [dist=2..3] & [div=0..5] V [neigh=5..6] & [genact=13..15] & [pcpt=1..2] V [day=0..7] & [hr=6..9] & [act=0..31] & [dist=1..2] & [div=2..8] V [hr=7..9] V [day=5..7]
:> [animal=1]

[hr=1..5] & [act=0..22] & [neigh=0..5] & [dist=0..3] & [div=0..1] & [sup=1..3] & [genact=0..19] & [hypact=2..5] V [hr=4..9] & [hum=0..5] V [hr=0..1] & [hum=6..7] V [day=1] V [day=3..8] & [hr=0..5] V [day=3..8] & [neigh=1] V [hr=6] & [act=2..40] & [neigh=6] & [dist=1..9] & [div=0..5] & [sup=1..9] & [genact=0..15] V [day=3..4] & [hr=0..6] & [act=14..40] & [dist=0..1] & [genact=5..26] V [hr=5..6] & [act=0..2] & [neigh=6..7] & [genact=2..19] V [hr=6] & [neigh=6] & [div=2..8] & [hypact=2..3] V [tem=0..6]
:> [animal=2]

[day=2..5] & [hr=0..7] V [day=6..8] & [hr=9] & [neigh=4..7] & [genact=0..15] V [neigh=4] V [hr=9] & [neigh=4..10] & [div=0..1] & [sup=0..7] & [tem=0..6] V [hr=9] & [div=5] & [hypact=0..3] & [pcpt=0] V [hr=9] & [neigh=4..7] & [dist=1..2] & [div=3..8] & [genact=3..26] V [hr=9] & [neigh=4..7] & [dist=0] & [tem=0..6] V [hr=9] & [neigh=4..7] & [dist=2] V [sup=5] & [hum=0..5] V [hr=9] & [neigh=6..7] & [div=2..3] & [hypact=2..3]
:> [animal=3]

[hr=0..8] & [div=0..6] & [tem=8] V [hr=0..8] & [dist=1..9] & [hum=0..6] V [hr=4..9] & [neigh=3] V [hr=7..9] & [neigh=8..10] & [tem=7..8] V [hr=6..9] & [neigh=7] & [div=4..8] & [genact=0..19] & [hypact=0..4] V [hr=7..9] & [act=4..40] & [tem=7..8] V [hr=5..7] & [neigh=7..10] & [pcpt=0] V [act=0..22] & [neigh=8..10] & [div=2..8] & [genact=3..26] & [pcpt=0] V [hr=7..8] & [div=0..4] V [hr=7..9] & [neigh=8..10] & [div=6..8] V [hr=6..9] & [neigh=5..10] & [dist=1] & [div=2] & [hypact=3..4] V [hr=8..9] & [genact=17..19] V [hr=9] & [act=4..30] & [dist=0] & [div=5..8] V [day=6..8] & [neigh=8..10] & [dist=1..9]
:> [animal=4]

Figure 27. Rules Derived by GEM from Monkey Data.

thresholds, evaluation schemes fall in to 3 classes: those with relatively high q-thresholds, those with medium q-thresholds, and those with low q-thresholds. This is a consequence of the inherent spectral assumption of our selected evaluation schemes: schemes at the high end of the thresholds allow a high q-threshold and those at the low end require a low q-threshold. A further classification was found with respect to to variance of the leading confidence values from rules: high variance vs. low variance. That is, certain schemes have high q-thresholds and low variance, others low q-thresholds and high variance, etc.. In fact, a heuristic seems to be that relatively high q-thresholds are better with wide variance and

low q-thresholds are worst with wide variance.

Rather than a pure gradient technique for converging on optimal values, a manually applied *hill climbing* method was used in the simulation experiments. Some of the results obtained for various performance metrics are summarized in Figure 28 and 29. The interesting outcome of this experiment is that the best case results of the simulation and the control cases differ slightly. For the simulation, the standard MIN/MAX, PSUM/AVE schemes gave good performance, but the control events indicated that AVE/RMS and ISD2/RMC to be more reasonable choices. Although the first-rank hits and first-only hits differ, the product of the two is more important, and it is the same.

The results of the control events are shown in Figure 28. These results were used to determine the q-threshold for the simulation ($\tau_1 = 0.96$) and to throw out parts of the AND/OR scheme spectrum which did not appear to be particularly interesting (high confusion, high off diagonal entries in the matrix, or low average confidence and variance of leading hypothesis). Although the simulation results for the corresponding AND/OR scheme of the control events was reasonably well behaved with respect to secondary statistics (e.g., confusion), the best results are shown in Figure 29. One must bear in mind that these results are less than impressive due to the fact that there was never any hope of perfecting this rule set since expert criticism and refinement of the rules was not available. Rather, the thing to be observed here was the *relative* performance of various evaluation schemes.

Thus, although the presence of *false positives* (i.e., confidence above threshold for a class in addition to the correct class) and *false negatives* (i.e., confidence above threshold for a class other than the correct class) can not be meaningfully addressed by this methodology, relative performance measures can be applied to the problem of "confusion" as defined earlier (i.e., the relative separation of confidence values). Furthermore, there are in fact other techniques for dealing with the debugging of false negatives and false positives which are discussed elsewhere in the literature for automated rule debugging.

For the AVE/MAX scheme, the results look considerablely better than for AVE/PSUM, and the MIN/MAX case looks quite as good as AVE/MAX. The off-diagonal average in AVE/PSUM is 83.8, and

Confusion Matrix				
Correct Classification	Assigned Classification			
	Animal1	Animal2	Animal3	Animal4
Animal1	100	50	26	88
Animal2	39	99	36	60
Animal3	100	98	66	53
Animal4	87	85	15	98

First Rank Hits = .851, First Only Hits = .53
Average Leading Confidence = 0.613 , Variance = 0.006, Range = [0.458,1.0]
Number of Control Events = 1815 , τ_1 = 0.45 , τ_2 = 0.15 , AND = "AVE" , OR = "RMS"

Confusion Matrix				
Correct Classification	Assigned Classification			
	Animal1	Animal2	Animal3	Animal4
Animal1	100	13	5	75
Animal2	60	89	0	39
Animal3	87	90	49	40
Animal4	78	58	22	91

First Rank Hits = .76, First Only Hits = .585, Expectation of Type I Confusion = 0.001
Average Leading Confidence = 0.638 , Variance = 0.018, Range = [0.444,1.0]
Number of Control Events = 1815 , τ_1 = 0.45 , τ_2 = 0.15 , AND = "ISD2" , OR = "RMS"

Confusion Matrix				
Correct Classification	Assigned Classification			
	Animal1	Animal2	Animal3	Animal4
Animal1	100	34	62	100
Animal2	68	100	50	92
Animal3	100	100	99	64
Animal4	96	86	23	100

First Rank Hits = .81, First Only Hits = .57
Average Leading Confidence = 0.69 , Variance = 0.456, Range = [0.45,1.0]
Number of Control Events = 1815 , τ_1 = 0.45 , τ_2 = 0.15 , AND = "ISD2" , OR = "RMC"

Figure 28. Results from Monkey Rules Control Events Evaluation.

Confusion Matrix				
Correct Classification	Assigned Classification			
	Animal1	Animal2	Animal3	Animal4
Animal1	100	94	74	97
Animal2	87	100	90	96
Animal3	85	94	100	95
Animal4	79	76	39	100

Expectation of Type II Confusion = 0.0031, Multiplicity = (0,0,0.077,0.92)
Average Leading Confidence = 0.866 , Variance = 0.033, Range = [0.2,1.0]
Number of Simulation Events = 20,000 , $\tau_1 = 0.97$, $\tau_2 = 0.50$, AND = "AVE" , OR = "PSUM"

Confusion Matrix				
Correct Classification	Assigned Classification			
	Animal1	Animal2	Animal3	Animal4
Animal1	100	93	72	64
Animal2	0	100	88	46
Animal3	0	0	100	35
Animal4	0	0	0	100

Expectation of Type II Confusion = 0.093, Multiplicity = (0.0022,0.039,0.155,0.80)
Average Leading Confidence = 0.866 , Variance = 0.033, Range = [0.2,1.0]
Number of Simulation Events = 20,000 , $\tau_1 = 0.97$, $\tau_2 = 0.50$, AND = "AVE" , OR = "MAX"

Confusion Matrix				
Correct Classification	Assigned Classification			
	Animal1	Animal2	Animal3	Animal4
Animal1	100	93	72	64
Animal2	0	100	88	46
Animal3	0	0	100	35
Animal4	0	0	0	100

Expectation of Type II Confusion = 0.092, Multiplicity = (1.0,0,0,0)
Average Leading Confidence = 0.615 , Variance = 0.237, Range = [0.0,1.0]
Number of Simulation Events = 20,000 , $\tau_1 = 0.97$, $\tau_2 = 0.50$, AND = "MIN" , OR = "MAX"

Figure 29. Results from Monkey Rules Simulation.

in AVE/MAX and MIN/MAX the value is 33.2. On second glance, we see that the number of confusion events for AVE/PSUM is about thirty times less. Moreover, the *multiplicity* statistic indicates that those few confusion events were for situations in which three or four of the four classes were indistinguishable. Thus, depending on priorities, the AVE/PSUM scheme might be better. There is also a considerable difference between AVE/MAX and MIN/MAX. The average confidence is lower and variance wider among leading hypotheses for the MIN/MAX scheme. This is generally less than desirable since typically having as many confidence values as high as possible is desirable, unless this causes other problems.

One might assume some sort of distribution of values of confidence, e.g. the Student t-distribution (unknown variance and mean of standard normal distribution) , for specifying a confidence interval (in the standard sense of the word), for the value of average and variance of the actual distribution based upon the estimating parameters collected in the simulation statistics. It should be kept in mind that the density of the confidence space typically varies in a manner which is very peculiar, and this approach may be quickly abandoned.

6.3. About PLANT/ds

Within the expert system PLANT/ds for diagnosis of soybean diseases in Illinois, two rule groups operate simultaneously. One rule set is derived through the painstaken effort of plant pathologists to encode their knowledge in the form of machine executable rules. The other rule set is the product of the automated induction program AQ11/GEM run on exemplary cases of the diseases [Ch 79, ChMi 80]. PLANT has enjoyed relatively high popularity among the agronomists exposed to the system, yet the current rules suffer from a number of maladies.

From a pragmatic basis, two problems are of concern :

1) A rule may fail to recognise a true instance of a disease (miss-hit), or

2) A rule all-too-freely declares that some disease other than the correct one is present (false alarm).

Often only the latter of these problems is noticeably present in later stages of rule base development. A

typical situation is: "System reports most probable occurrence of disease = A with evidence that B, C, D, and E are present, B being most likely alternative and E being least likely. However, the expert making the inquiry states that disease C is the correct diagnosis, and moreover, that B is completely erroneous."

The original confusion matrices for the PLANT machine and expert rules, summarizing some 340 test cases, showed this problem rather dramatically. The fact that the diagonal entries of the matrix were almost solidly 100% for both human and machine rules attested to the skill of knowledge engineer and prowess of Aq11 respectively. Conversely, relatively high numbers in the off-diagonal entries of the table evinced the severity of problem (2). This syndrome which can be loosely labeled as "confusion" can be further conceptualized as :

a) assigning diagnoses to diseases on the basis of evidence which is too weak, or

b) assigning diagnoses to too many diseases at once.

Although the two overlap to a large degree, there is some distinction. The first case amounts to "jumping the gun" due to excessive sensitivity in evaluation, and the second case involves a lack of proper discrimination/filtering between the diseases either by the intrinsic nature of the disease or lack of proper granularity in the rules. Assuming that there is a solution, in the first case we can simply adjust the evaluation parameters, but in the other case we need to revise the descriptors and/or the rules.

6.4. PLANT Results

Since the experts who devised the original rules are no longer accessible on a regular basis, the goal of this experiment was to achieve the lowest number of (2) problems with the highest figures possible on the main diagonal. A fresh set of 289 control events was used for primary testing, and random events simulation was used to support this evidence (it would not be reasonable to trust the results of the primary experiment if the simulation runs indicated other problems, but this was not the case).

Figure 30 shows the attributes used in the PLANT system. The goal variable is *disease* which represents an assignment of a diagnosis from one of the 17 soybean diseases. There are 50 attributes, and one goal variable. In PLANT, any variable may be assigned the values of *UNKNOWN* (plausible propagation through any rules involving this variable) and *DOES NOT APPLY* (systematic failure of rules having this variable), but these values are not used in this experiment since the effect of incomplete

Attribute	Number of Values
time_of_occurrence	7
precipitation	3
temperature	3
cropping_history	4
damaged_area,severity	5,4
plant_height	2
condition_of_leaves	2
leaf_spots,leaf_spot_color,color_of_spot_on_reverse_side	3,12,12
yellow_leaf_spot_halos,leaf_spot_margins	3,3
raised_leaf_spots,leaf_spot_growth,leaf_spot_size	3,7,4
shot_holing,shredding	3,3
leaf_malformation	3
premature_defoliation,leaf_withering_and_wilting	4,3
leaf_mildew_growth	4
leaf_discoloration	7
position_of_affected_leaves	4
condition_of_leaves_below_affected_leaves	5
condition_of_stem	2
stem_lodging	3
stem_cankers,canker_margin	7,3
fruiting_bodies_on_stem	4
external_decay_of_stem	4
mycelium_on_stem	3
external_stem_discoloration,internal_discoloration_of_stem	7,5
location_of_stem_discoloration	4
sclerotia_internal_or_external	3
condition_of_fruit_pods	2
fruit_pods,fruit_spots	4,5
condition_of_seed	2
seed_mold_growth	3
seed_discoloration_color	7
seed_size,seed_shriveling	3,3
condition_of_roots	2
root_rot,root_galls_or_cysts,root_sclerotia	3,3,3
disease	17

Figure 30. Attributes and Values for PLANT Data.

data was not a concern for this experiment (generally, when flaws are found with rules in the handling of incomplete data, an expert is required for detailed analysis).

Figure 31 depicts a few typical rules from the PLANT system. These rules have been influenced by the involvement of 4 different experts and span the course of 6 years and are quite simple compared to some of the intermediate generations of rules in the PLANT system. There are also several *restriction* rules in the PLANT system which serve as background knowledge and enforce some domain specific

---

```
  [canker_lesion_color = reddish_brown]:0.70
V [canker_lesion_color = brown]:0.40
V [discoloration_external = brown]:0.40
V ([stem_cankers = near_soil_line] V [location_of_discoloration = near_soil_line]):0.50
V [root_rot = present]:0.20
V [stem_condition = abnormal]:0.10
        :>   [disease = rhizoctonia]


  [time_of_occurrence = august..october]&[growth_leaf_spots = from_edge_of_leaf_inward]:0.75
V [internal_discoloration_of_stem = brown]:0.50
V [leaf_discoloration = yellowed_from_margins_or_interveinal]:0.50
V [size_of_leaf_spots = greater_than_eighth_inch]:0.20
V [leaf_withering_and_wilting = present]:0.20
V [leaf_spots = present]:0.10
V [leaf_condition = abnormal]:0.05
        :>   [disease = brown_stem_rot]


  [time_of_occurrence = august..october]&[terminal_buds = curved]:0.90
V [time_of_occurrence = august..october]&[plant_height = less_than_normal]:0.80
V [time_of_occurrence = august..october]&[bud_necrosis = present]:0.70
V ([fruit_diseased = underdeveloped] V [fruit_spots = dark_blotches]):0.70
V [plant_color = green_after_maturity]:0.70
V [time_of_occurrence = august..october]&[leaf_malformation = present]
  &[leaf_discoloration = bronze]:0.50
V [internal_discoloration_of_stem = brown]:0.40
V [leaf_brown_streaks = present]:0.30
        :>  [disease = bud_blight]
```

Figure 31. Expert Rules From PLANT/ds Expert System.

---

conditions, but these are not needed with the control events; rather, only with the simulation runs. A typical restriction rule is:

[condition_of_leaves = normal]
→
[leaf_spots = absent]
[leaf_malformation = absent]
[leaf_discoloration = absent] .

A preliminary run over the 289 control events with very low thresholds ($\tau_1 = 0.05$, $\tau_2 = 0$) using a wide spectrum of AND/OR possiblities was run initially to explore the q-thresholds. Subsequently, thresholds were set accordingly and additional runs determined that {AND = "AVE" , OR = "PSUM"}

Confusion Matrix																	
Correct	Assigned Diagnosis																
	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17
D1	94	6	82	82	0	0	0	0	0	0	0	71	0	0	0	24	0
D2	41	94	35	29	0	0	0	0	0	0	0	53	6	0	0	35	0
D3	65	76	100	71	0	0	0	0	0	0	0	59	0	0	0	0	0
D4	35	71	94	82	53	0	0	0	0	0	0	0	0	0	0	0	0
D5	82	0	0	24	88	0	0	0	12	0	0	0	35	35	0	0	0
D6	12	0	0	0	0	100	0	0	6	0	0	6	29	0	0	0	0
D7	0	0	0	0	6	0	94	29	6	6	0	0	53	6	6	0	0
D8	53	0	47	53	0	0	0	100	76	12	0	41	82	0	29	0	0
D9	0	0	0	0	0	0	0	100	100	0	0	24	41	29	53	0	0
D10	0	0	0	0	0	0	0	100	94	88	0	6	71	18	53	0	6
D11	0	0	0	0	0	0	0	12	0	0	94	35	18	0	6	0	0
D12	100	0	24	24	47	0	0	0	0	18	0	100	18	47	0	18	0
D13	0	0	0	0	76	0	0	29	0	0	0	6	100	41	6	0	0
D14	0	0	0	0	94	0	0	0	100	0	0	100	100	100	94	0	0
D15	47	0	24	29	53	0	0	53	24	0	0	41	59	41	82	0	0
D16	82	0	0	0	0	0	0	0	0	0	0	82	0	0	0	82	0
D17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100

First Rank Hits = 0.80, First Only Hits = 0.74
Average Leading Confidence = 0.893 , Variance = 0.0088, Range = [0.55,1.0]
Number of Control Events = 289, $\tau_1 = 0.55$, $\tau_2 = 0.20$, AND = "AVE" , OR = "PSUM"

Figure 32. Results from PLANT Rules Evaluation Using AVE/PSUM.

and {AND = "IDSP2" , OR = "PSUM"} were the best results, with q-threshold of 0.54. These results are shown in Figures 32 and 33.

Finally, a simulation run over the most prominent AND/OR schemes (AVE/PSUM, MIN/PSUM included) was used to confirm the stability of these results for incomplete data. Figure 34 gives the results of the simulation experiments using 20000 events. The AVE/PSUM combination was most interesting since the overall confusion expectation was lowest among the possibilities suggested by the control runs. This is a very satisfying result since it reaffirms earlier work by Reinke [Re 84] showing AVE/PSUM to be a proper choice for the PLANT/ds rule base. The IDSP2/PSUM scheme also showed

Confusion Matrix																	
Correct	Assigned Diagnosis																
	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17
D1	94	6	82	82	0	0	0	0	0	0	0	71	0	0	0	24	0
D2	41	94	35	29	0	0	0	0	0	0	0	53	6	0	0	0	0
D3	65	76	100	71	0	0	0	0	0	0	0	59	0	0	0	0	0
D4	35	71	94	82	53	0	0	0	0	0	0	0	0	0	0	0	0
D5	82	0	0	24	71	0	0	0	12	0	0	0	35	6	0	0	0
D6	12	0	0	0	0	100	0	0	0	0	0	0	29	0	0	0	0
D7	0	0	0	0	0	0	94	6	6	6	0	0	53	0	0	0	0
D8	53	0	47	53	0	0	0	65	76	12	0	29	82	0	0	0	0
D9	0	0	0	0	0	0	0	94	100	0	0	0	41	6	6	0	0
D10	0	0	0	0	0	0	0	88	94	88	0	0	71	12	18	0	6
D11	0	0	0	0	0	0	0	0	0	0	94	0	18	0	0	0	0
D12	100	0	24	24	0	0	0	0	0	18	0	82	18	0	0	0	0
D13	0	0	0	0	18	0	0	6	0	0	0	0	100	6	6	0	0
D14	0	0	0	0	0	0	0	0	100	0	0	0	100	100	94	0	0
D15	47	0	24	29	0	0	0	24	6	0	0	12	59	18	82	0	0
D16	82	0	0	0	0	0	0	0	0	0	0	29	0	0	0	82	0
D17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100

First Rank Hits = 0.78, First Only Hits = 0.72
Average Leading Confidence = 0.886 , Variance = 0.01, Range = [0.54,1.0]
Number of Control Events = 289, $\tau_1 = 0.54$, $\tau_2 = 0.20$, AND = "MIN" , OR = "PSUM"

Figure 33. Results from PLANT Rules Evaluation Using MIN/PSUM.

results similar to the AVE/PSUM , only marginally less : the confusion expectation was 0.01 compared to 0.0063 . Considering the fact that the IDSP2 scheme is more desirable from an efficiency view, this is probably a better choice, despite the slightly higher confusion potential.

Confusion Matrix																	
Correct	Assigned Diagnosis																
	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17
D1	100	7	8	16	16	0	0	11	11	0	0	31	21	13	4	0	4
D2	27	100	30	46	22	17	12	26	26	19	12	37	21	17	3	15	11
D3	24	24	100	40	24	0	0	17	10	1	0	22	15	7	1	0	6
D4	28	21	30	100	32	7	7	23	25	12	5	43	14	15	3	9	8
D5	17	33	14	34	100	0	0	20	16	0	0	33	34	20	2	2	2
D6	24	27	29	51	31	100	0	24	27	15	2	35	19	13	3	5	5
D7	25	25	26	43	27	0	100	26	30	11	0	34	20	12	2	9	3
D8	37	13	22	38	19	0	0	100	67	9	0	43	4	1	1	0	5
D9	21	18	12	16	8	3	3	28	97	7	0	42	0	0	0	0	0
D10	23	31	26	47	29	13	13	40	34	100	4	41	20	14	2	11	7
D11	25	35	26	49	20	15	17	27	29	22	100	40	19	19	3	17	15
D12	32	27	20	39	23	6	8	18	30	7	8	100	16	17	5	4	9
D13	27	24	24	25	38	7	3	8	11	9	0	31	100	15	0	3	0
D14	22	33	29	47	26	11	13	20	22	13	7	39	13	100	4	10	6
D15	27	38	28	56	25	14	16	26	28	24	20	40	16	21	100	18	21
D16	23	28	25	47	29	15	15	22	26	13	1	34	15	17	2	100	6
D17	27	40	26	55	22	13	17	28	27	21	14	42	18	22	6	17	100

First Rank Hits = .994, First Only Hits = .70
Expectation of Type II Confusion = 0.0063, Multiplicity = (0.03,0.04,0.03,0.07)
Average Leading Confidence = 0.919 , Variance = 0.0057, Range = [0.45,0.99]
Number of Simulation Events = 20,000 , $r_1 = 0.54$, $r_2 = 0.20$, AND = "AVE" , OR = "PSUM"

Figure 34. Results from PLANT Rules Simulation.

CHAPTER 7

OBSERVATIONS AND CONCLUSIONS

7.1. Discussion

Confusion arises at several levels, but a root cause seems to be the assignment of confidence values to the partial evaluations of rules which have been misguidedly forced to be homogeneous (unduly uniform, lacking sufficient diversity in choice of evaluation schemes). Further, expert rules are derived under somewhat ill–defined pretexts perhaps. When the human comes up with the rule, he is not aware of the consequences of the machine's evaluation of his rule (e.g., consequences of partial satisfaction of rules). He is only considering the relevant variables in totality, not the exception driven cases necessary for accurate partial evaluation. There is a violation of knowledge granularity between man & machine since the man assumes (upon the recommendation of the intrepid knowledge engineer) that if he only works at the highest level of his experience (ie., complete knowledge), he can trust the machine to accomplish the task of exception handling (which he has built into his own knowledge base) in a manner similar to that which he himself would have used. By this, one effects results in the machine evaluation scheme that are quite different than anticipated. Moreover, since the inherent assumption is relatively easy to make and somewhat obscure, another expert will readily verify the rectitude of the first expert's rules but will fail just the same to see the reason for the rules not stacking up to expectations.

These premises argue towards 3 possible improvements in technique:

a) better human formulation of rules for handling incomplete knowledge
and better man–machine interface,

b) better induction algorithms which can develop rules stable with respect to incomplete data,

c) better testing methods which can not only evaluate rules but also find
appropriate plausibility schemes.

7.2. Better Rules

One problem, which is more a fault of the expert than the machine, is that there does not seem to be a well balanced use of descriptors in the expert rules. Since the expert himself came up with the descriptors, it is unlikely that the unused descriptors actually are irrelevant. They are simply irrelevant to this particular expert in the context of his own problem solving method at a particular point in time. Somehow he knows that he is still learning or perhaps he also realizes or others have pointed out the significance of values other than the ones he is presently using. A solution may be to ask several other experts to augment the rules to try and add supporting cases and exceptions to our existing expert rules. In effect, this integrates multiple sources of knowledge [Michie]. Whatever, inbreeding of expert knowledge should be avoided. Diversity in experts will build a stronger system.

A prime deficiency of PLANT is its inability to utilise more than one evaluation scheme within a rule group. This disallows simultaneously assessing quasi-optimal diagnosis plausibility of individual classes of diseases. E.g., it was noted that the best evaluation scheme for stem diseases differs from the scheme for leaf spot diseases, but within a rule group only one evaluation scheme is allowed. Given sufficient real world data, this presents no problem to a closed system model, but sparse or incomplete field data (especially in the PLANT/ds event space of size $2x10^{30}$) mandates optimal use of the user's input. Consequently, a non-homogeneous evaluation is proposed to alleviate the problem. Non-homogeneity can be effected by imposing a supervisory control mechanism above the existing evaluation schema, which will ultimately dispatch the proper subgroup of rules once sufficient evidence precludes satisfaction of any other subgroup. Alternatively, a special rule group to accomplish rule group switching might be defined to direct the system to the proper subgroup with its associated evaluation scheme. Note that this partitioning of rules into rule groups is strictly a performance issue, but semantically meaningful aggregates of rules might also be a basis for rule groups. The current ADVISE [Baskin] implementation of PLANT is particularly conducive to this approach. Further, the facilities in ADVISE, an "empty expert system", allow a wealth of other questions regarding evaluation schema to be

quickly and painlessly determined.

7.3. Better Induction Algorithms

Surprisingly, humans possess an uncanny proclivity for exception handling. Children of age 2–7 encounter no difficulty in breaking down their internalized models of language and mores and physical reality to rebuild them in the light of conflicting environments until a model consistent with their surroundings is achieved. We continue this process throughout our adult life, but with a different emphasis and motivation. Exceptions trigger demon processes so subtly that often we are not aware of any exceptional condition having occurred. This is the true power of human thought. It no longer becomes necessary to have complete models when the exception handling is well integrated.

Regardless of quality of rules and scheme for evaluating them, a machine needs some way of finding exceptions and resolving them in a systematic way. The simulation approach advocates an attempt to find a lot of data that could be found by computational methods, but the computational methods are too time consuming in terms of real man-hours. Some of the exceptions might be found in this way before they come up in the real life of the system (interactive on-line learning sessions with humans). There is no guarantee that the machine can resolve these exceptions in terms of the tools/resources available (an expert may be available to modify the rules, an incremental learning induction program may be available for exception resolution), but at least it "knows" about them. This is all humans can do at times to cope with exceptions: simply to be aware of them as potential problem areas.

Past work by Michalski and Reinke [MiRe 84] has suggested various uses of a *voting* algorithm for fixing weights in an induction process. This notion may be of some use in making minor modifications to the resultant rules or integrating the simulation process model into an induction algorithm to provide additional degrees of freedom in the interpretation of the rule evaluator output.

First let us consider the basic idea for a learning algorithm. Selectors and groups of selectors are assumed to have a variable associated with a weight pertinent to the variable's or group's context in the

rule for a specific classification (disease). This variable starts at some reference value (say 0) and is incremented or decremented according to the importance of the variable in a particular example and the utility of consolidating the variable into a description (incipient rule) for the class to which it belongs. When all exemplars have been processed, the weight variables are normalised and installed into the logic rules to form decision rules.

Now consider saving the state of the weighting information (not only the weights, but actually subtallies of how many times a variable value contributes to the unique and non-unique satisfaction of individual rules). In the course of testing, this information can easily be updated and new weights derived. This nicely handles the desire to make a rule acquisition system adaptive with respect to a changing environment (changing source of rules, new and better source of examples, new/special cases), and constitutes a variant of online incremental learning. Thus, in essence a control scheme to handle partial (incomplete) data and other plausibility aspects can be built incrementally.

Although it it not appropriate to overly criticise the the *monkey rules* toy problem, several problems with the PLANT/ds rules are noted beyond the strict choice of AND/OR evaluation schemes, particularly with regard to contexts for selector and selector groups. One shortcoming is a sparse use of weight functions or weight values within selectors to establish the proper "context" for a selector or group of selectors. Although some work has been done on the automatic induction of weights [MiRe 84], the experts manually assigned weight values to complexes. This is in fact a tiresome and tedious task accomplished through trial and error. One accomplishs this somewhat heuristically by default in the present case, but one might assume an increasing wealth of knowledge being accumulated in an online learning environment for the proper weighting of terms based on an expert's interaction with the system. Now the particular partial evaluation function for a term might likewise be learned in this manner. Each term could be a probabilistic memo function with two alternatives as the learning set :

i) accumulation of end user statistics as to number of times a specific selector is set to a value in a particular context, so as to build

an ad hoc probability distribution which would become more accurate with time as more and more users used the system,

ii) accumulation of expert consulting time to adjust weights and thresholds in the complete knowledge of all past test cases,

or alternatively perhaps these two techniques could be hybridized. The first involves learning by interaction with a certain segment of the the real world, namely farmers or county agents. The second involves learning from experts. In any case, more work needs to be done in balancing the weights and confidence coefficients (necessary & confirmatory) under automated control, preferably in a learning loop.

7.4. Integrating Testing and Learning

The consolidation of the reference mode and stochastic mode operation of the current rule evaluator could easily be implemented as a utility basis feature for an advanced induction algorithm. As previously mentioned, the evaluator provides metrics pertinent to both the completeness and the consistency of a rule base. The statistics are quite useful prompts for various induction procedures. In reference mode, particular control events can be identified as performing poorly, and in stochastic mode, particular exception events (*most characteristic* exception events if too numerous) can be the emphasis of incremental learning.

There are other instances in which the induction process might be driven or assessed under the control of a joint reference/stochastic mode evaluator, even for partially constructed rules. The importance of both modes in the evaluator can not be stressed enough. The reference mode operation gives some feedback on the performance of the rules against known test cases. It further serves as a consistency check against any changes prompted by the more theoreticly based stochastic mode. The stochastic mode operation explores regions of the event space that are perhaps not foreseen in the construction of the "typical" test cases. It gives the characteristics of the rules in the more pragmatic sense of the real world by simulating imprecision, unreliability, and incompleteness. Further, it provides

some hints of ways that the rules could be better in the more general sense of utility over the whole event space (not just the restricted space of the learning set).

7.5. Conclusions

Work on the ultimate incorporation of the exerciser into the ADVISE Meta– Expert System [MiBa 84] has already begun. The goal is an integrated system for the automatic generation of expert systems. Future work will incorporate a more interactive, user-friendly rule testing program into a window-based, multitasking environment capable of interaction with other modules : rel.database, rule parser, inductive learning algorithms, and end-user consultations. However, the need for an alternate (perhaps variantly specialised) rule representation scheme in ADVISE is required to accommodate REAP. Figure 35 illustrates the difference in overall execution time between the currently interpretive execution of rules in ADVISE versus the compiled and executed rules of REAP.

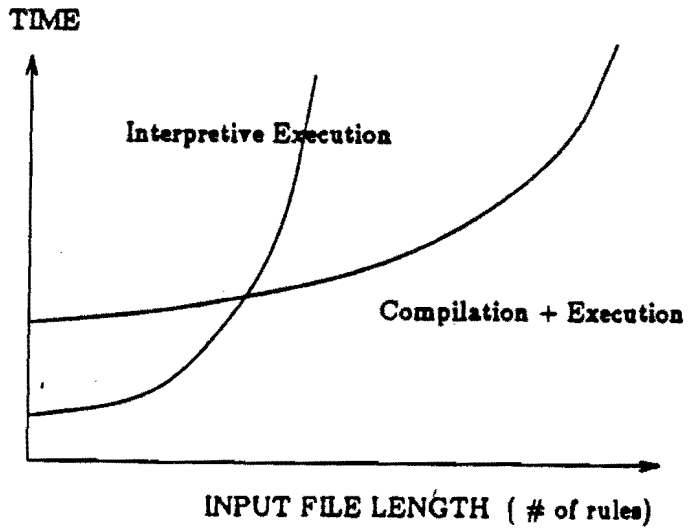


Figure 35. Typical Tradeoff of Interpreted vs. Compiled Rules

Since computation efficiency is a desired performance measure of most expert systems, a number of computational short–cuts should be built into the rule evaluator mechanism. Some of the ideas yet to be exploited which have been hinted at herein are listed below.

> 1) Normalize confidence values to the range 1 - 1000 to correspond to the interval [0.001,1.00]. This allows several computational techniques such as right shift in place of divide by two. All the number can be reconverted to the proper range on printout, but for the scores of intermediate computations, integer arithmetic is more efficient .
> 2) Use ID2S wherever possible in place of AVE.
> 3) Use limited precision computation, particularly for weights, where appropriate.

The cost tradeoffs between completeness and consistency can be built into the utility function $U(C_{consistency}(X), - C_{completeness}(X))$ along with other performance characteristics. E.g., ISD2 is almost as cheap in terms of computation time as MAX or MIN or AVE. Thus, one might throw a term in the utility function to take this into account even if completeness alone is not optimal..

Due to the CPU intensive nature of the simulation technique, a rampant gradient search directed program run may not be such a good idea. Rather a manually executed schedule of runs may be evaluated by a humanly obviated algorithm which is not well defined. It should be the goal of future research to clarify the algorithm so as to allow secure automated execution of the search space.

Note that an advanced confidence generating model was not employed, so various schemes aforementioned would be quite identical. A model employing restriction rules and probabilistic confidence levels from some sort of concurrent simulation processing would be a logical next step for experiments.

The use of multiple experts is encouraged in this methodology. In Figure 1, this is represented by the two arrows emanating from the "examples oracle". In fact, it is best to try to simulate 2 separate oracles, even when only a single expert is available.

Both the MONKEY rules and the PLANT rules involve involve complications that would make direct verification of consistency and completeness difficult. In the MONKEY rules, the length of the rules are the problem, and in the PLANT rules the weights are the problem. Both rules turn out to have rather complicated plausibility propagation schemes.

Efficiency of evaluation is a concern in most rule based systems because it is necessary to repetitively reevaluate the status of rules, and there are quite typically hundreds of rules. The use of short integers and short precision arithmetic suits the need for 2 or 3 significant figures for "confidence values" in such circumstances. Tables of logs for exponentiation and multiplication make the less conventional plausibility schemes more feasible. Use of fixed point arithmetic allows use of the boolean shift operation in lieu of division in the IDS2 scheme which showed great potential. Likewise, weights are not exactly precise, and wide variance in weight values may be tolerable in the interest of more efficient computation.

The statistics that show most promise for future work are

first rank percentages

first only percentages

the overall confusion statistic

confusion matrix average main diagonal entry

confusion matrix average off diagonal entry.

For the PLANT and MONKEY rules, the standard schemes performed reasonably well, but more significant is the fact that less standard schemes, such as IDS2 performed reasonably. The IDS2 scheme is potentially computationally more efficient than AVE, and hence important. The RMS scheme did significantly better in MONKEY rules which suggests that AVE/PSUM and MIN/MAX are only starting points in a search for better rule performance.

REFERENCES

[Am 71] Amstadter, B. L., **Reliability Mathematics**, McGraw–Hill Book Co., New York, 1971.

[Ba 83] Baim, P.

[Br 79] Bratko, I., **A An Advice Program for the King–Knight vs. King–Rook Ending**, Research Memorandum MIP–R–123, Edinburgh: Machine Intelligence Research Unit, University of Edinburgh, 1979.

[Ch 79] Chilausky, R.L., **A Prototype Computer Based Consulting System Using the Variable Valued Logic System VL1 : Methodology and Implementation**, M.S. Thesis, Department of Computer Science, University of Illinois, Urbana, January, 1979.

[ChMi 80] Chilausky, R.L. and R.S. Michalski, **Knowledge Acquisition by Encoding Rules versus Computer Induction from Examples : A Case Study Involving Soybean Pathology**, *International Journal for Man–Machine Studies*, No.12, 1980, pp.63–87.

[Du 79] Duda, R.O., Hart, P.E. and Nilsson, N., **A Computer–Based Consultant for Mineral Exploration**, Final Report AER 77–04499, SRI International, 1979.

[Fr 80] Friedman, L., **Reasoning by Plausible Inference**, *Proceedings of the Fifth Conference on Automated Deduction*, Les Arcs, France, July 1980.

[Fr 81] Friedman, L., **Exteneded Plausible Inference**, *Seventh International Joint Conference on Artificial Intelligence*, University of British Columbia, Vancouver, August, 1981.

[Ga 81] Garvey, T.D., **An Inference Technique for Integrating Knowledge from Disparate Sources**, *Seventh International Joint Conference on Artificial Intelligence*, University of British Columbia, Vancouver, August, 1981.

[GoNg 85] Goodman, I.R., Nguyen, H.T., **Uncertainty Models for Knowledge Based Systems**, North–Holland Publ., New York, 1985.

[Ha 79] Hayes, P. and Reddy, R. *An Anatomy of Graceful Interaction in Spoken and Written Man–Machine Communication*, Dept. of Computer Science, Carnegie–Mellon University, Internal Report CMU_CS_79–144.

[Jo 79] Johnson, Stephen C., **YACC: Yet Another Compiler–Compiler**, Bell Laboratories, Murray Hill, N.J., 1979.

[Kr 68] Kreiling, Frederick C., **Leibnis**, *Scientific American*, vol. 218, No. 5, pg. 95, New York, May, 1968.

[La 78] Larson, J.B. and R.S. Michalski, **Selection of Most Representative Training Examples and Incremental Generation of VL_1 Hypotheses: The Underlying Methodology and**

Descriptions of Programs ESEL and AQ11, Report No.877, Department of Computer Science, University of Illinois, Urbana, May, 1978.

[Mi 74] Michalski, R.S., **A Variable-Valued Logic : System VL_1,** *Proceedings of the Fourth International Symposium on Multi-Valued Logic*, West Virginia University, Morganstown, May, 1974, pp.323-346.

[MiBa 83] Michalski,R.S. and Baskin,A. B., **Integrating Multiple Knowledge Representations and Learning Capabilities in an Expert System: the ADVISE System,** *Eighth International Joint Conference on Artificial Intelligence*, pp.256-258, 1983.

[MiBa 84] Michalski,R.S. and Baskin,A. B., Seyler, M.R., Boulanger, A.G., **A Technical Description of the ADVISE Meta-Expert System,** Internal Report, Intelligent Systems Group, Department of Computer Science, University of Illinois, 1984.

[MiDa 83] Michalski,R.S., Davis, J.H., Bisht, V.S., and Sinclair, J.B., **A Computer-Based Advisory System for Diagnosing Soybean Diseases in Illinois,** *Plant Disease*, April, 1983.

[MiRe 84] Michalski.R.S., Reinke,R.E., *GEM Users Manual*, Internal Report, Intelligent Systems Group, Department of Computer Science, University of Illinois, 1984.

[Mic 80] Michie, D., **Expert Systems,** *The Computer Journal*, Vol.23, No.4, pg.369, January, 1980.

[Min 63] Minski, M., *Steps toward Artificial Intelligence*, Feigenbaum and Feldman (Eds.), **Computers and Thought,** New York, McGraw-Hill, 1963, pp. 406-450.

[Mit 64] Mitrinovic, D., **Elementary Inequalities,** Groningen, The Netherlands, P. Noordhoff Ltd., 1964, pp. 1-30.

[Mit 70] Mitrinovic, D., **Analytic Inequalities,** Springer-Verlag, New York, 1970, pp. 75-80.

[Qu 82] Quinlan, J.R., **INFERNO: A Cautious Approach to Palusible Inference,** Internal Report, Rand Corp., 1982.

[Re 83] Reinke, R.E., **PLANT/ds: An Expert System for Diagnosing Soybean Diseases Common in Illinois (User's Guide and Program Description),** Report No. UIUCDCS-F-83-911, Department of Computer Science, University of Illinois, 1983.

[Re 84] Reinke, R.E., **Knowledge Acquisition and Refinement Tools for the ADVISE Meta-Expert System,** Report No. UIUCDCS-F-84-921, Department of Computer Science, University of Illinois, 1984.

[Shaf 76] Shaffer, G., **A Mathematical Theory of Evidence,** Pinceton University Press, 1976.

[ShBu 75] Shortliffe, E.H. and Buchanan, B.G., **A Model of Inexact Reasoning in Medicine,** *Mathematical Biosciences 23*, 1975, pp. 351-379.

[St 84] Stepp, R., **CLUSTER/paf,** PhD Thesis, Report No. UIUCDCS-F-84-xxx, Department of Computer Science, University of Illinois, 1984.

[Su 82] Suwa, W., Scott, A.C. and Shortliffe, E.H., **An Approach to Verifying Completeness and**