A VARIABLE PRECISION LOGIC INFERENCE SYSTEM EMPLOYING THE DEMPSTER–SHAFER UNCERTAINTY CALCULUS

BY

PETER FAHRID HADDAWY

B.A., Pomona College, 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, 1987

Urbana, Illinois

ACKNOWLEDGEMENTS

I wish to thank my thesis advisor, Professor Ryszard Michalski, for his support and guidance. He was a constant source of ideas and motivation.

I am grateful to Jim Kelly for his time and effort in developing the ESTIMATOR system and his patience in working with some early and very buggy versions of the system. I thank Professor Tamer Basar for taking the time to help me through some of the finer points of probability theory and for pointing out flaws in one of my early Bayesian inference schemes. John Wiegand helped implement an early Bayesian version of the VPL system. Brian Falkenheiner clarified some of the mysteries of Dempster-Shafer theory. Jeff Becker introduced me to the idea of typed logic. Krish Purswani suggested the Dempster-Shafer example. I would like to thank Alex Krowitz for his careful listening and astute observations; he helped me out of some real tight spots. I am thankful to the members of the ISG group for creating a friendly and stimulating research environment.

I am especially thankful to Lisa Wolf for her support and understanding, and for proof reading several drafts of this thesis.

This work was supported in part by the Office of Naval Research under grant N00014-82-K-0186, in part by the National Science Foundation under grant NSF DCR 84-06801, and in part by the Defense Advanced Research Project Agency under grant N00014-K-85-0878.

TABLE OF CONTENTS

1. INTRODUCTION

People are capable of making decisions under extremely adverse conditions, including situations in which information is incomplete or time is limited. A detective forms hypotheses concerning the identity of the killer in a murder case by piecing together evidence which is usually incomplete and uncertain. A race driver must make split second decisions in order to react quickly to a rapidly changing environment. One reason that people are able to tailor their reasoning to fit the demands of their environment is that they can both reason under limitations of resources available for solving a problem and they can reason in the face of uncertainty. This combination is crucial since resource limitations put constraints on the information which can accurately be gathered. Two abilities are important in this kind of decision making. First is the ability to decide which information to attempt to gather. Second is the ability to reason in the absence of some information and in the presence of information which could only be approximately ascertained.

The most common type of resource limitation faced by reasoning systems is limited time. One attempt at solving the problem of real time response has been to make inference engines run faster. While such efforts are laudable in their own right, they do not properly address the problem of real time inference. With this approach, the time constraints of the application must be known before the knowledge based system is designed to insure that inference times are within limits. This puts unnatural constraints on the knowledge base design and makes for a very brittle system. The knowledge engineer is not free to encode the knowledge in the way that most closely reflects the expert's knowledge. Instead, he must keep the length and breadth of inferences within certain limits to guarantee required response time. There is no simple way to add new knowledge to such a system. In addition, in situations where the system has more than the minimum time limit to perform an inference, there is no way of using the additional time to improve the results.

The research presented in this thesis addresses the problem of reasoning under time constraints with incomplete and uncertain information. It is based on the ideas of Variable Precision Logic, introduced by Michalski & Winston [1986]. The approach taken is to vary the precision of inferences in order to produce the most accurate answer possible within a given time limit. This method produces a highly flexible system. Information can be added to the knowledge base without undue concern for inference times. In addition, the possible range of required inference times does not need to be known before a knowledge base can be encoded. The system can simply adjust to the amount of time available.

The problem of reasoning under time constraints with incomplete and uncertain information is closely related to the problem of reasoning efficiently with exceptions. Solutions to the latter problem shed light on the former one. Recent work in machine learning has addressed the problem of adding knowledge to existing knowledge bases, also known as knowledge assimilation. When new information is obtained, this information must be integrated with the existing knowledge in a consistent manner. There are two approaches to assimilating new knowledge: knowledge evolution and knowledge revolution [Michalski, 1985]. Revolution refers to formulation of completely new hypotheses, while evolution refers to modification of the existing hypotheses. In general, the revolutionary approach is far more computationally costly than the evolutionary approach. Thus the evolutionary approach is preferable in situations where the amount of time required to learn a concept is important, e.g., in dynamic environments where the learning process must keep pace with the rate by which information is gathered.

The current research complements one particularly promising method of evolutionary learning called exception learning. Under this methodology, rules are learned which apply to a general class of situations. Particular situations to which they do not apply are noted as exceptions. Specifically, when a rule is found to produce a false positive classification, the misclassified case is labeled as an exception to the rule. Two systems capable of learning exceptions are Excel [Becker, 1985] and Winston's learning system [Winston, 1986]. Accumulation of exceptions is a highly efficient learning method; however, the rules created rapidly become branchy, leading to inefficient reasoning. The Variable Precision Logic reasoning methodology is one way of reasoning efficiently with such exception augmented rules.

2. THEORY OF VARIABLE PRECISION LOGIC

Variable Precision Logic deals with both the problem of time constrained uncertain reasoning and the problem of reasoning efficiently with exceptions. In so doing, it displays an adaptability of reasoning behavior which until now has been observable only in human inference. This section discusses the theory of Variable Precision Logic as presented by Michalski & Winston [1986].

2.1. Time, Certainty, & Specificity

In any practical reasoning process there are extra–logical cost constraints such as time and resource limitations which must be taken into account. These constraints are generally dictated by the requirements of the particular environment in which the reasoning agent is functioning. In order to flexibly respond to these constraints, the tradeoff between the cost,

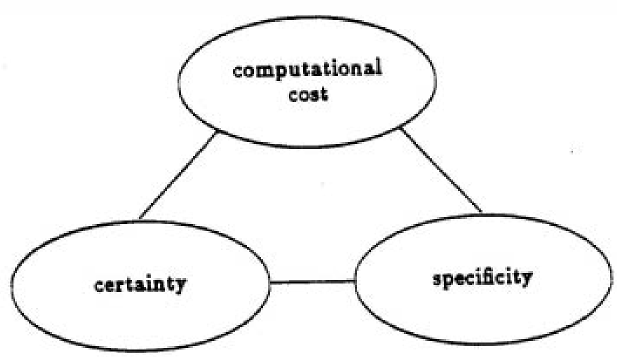


Figure 1. Triad of Tradeoffs*.

* Introduced by Michalski at the ISG seminar, October 1985.

certainty, and specificity of inferences can be used (see figure 1). In this context, certainty refers to the degree of belief in a statement, while specificity refers to the degree of detail of a description.

As an example of these tradeoffs in reasoning consider the following. Suppose you are standing in front of the Sears Tower with a friend. You ask him, "Are there more than one hundred offices in the building?" and he answers very quickly, "Yes." You ask him "Are there between one hundred and five thousand offices?" and he takes a bit more time. Finally you ask him "Are there between one and two thousand offices in the building?" and he takes much longer to respond. Each question requires progressively more deliberation. This is an example of the tradeoff between specificity and time. The more specific or narrow the probe question, the more time it takes for a response.

Now suppose that your friend is not required to answer only yes or no but can provide approximate answers. You ask him "Are there between one and two thousand offices in the building?" and he answers immediately "Maybe, maybe not." You tell him that you want more commitment than that and he replies "Most likely not." Finally, you tell him to either answer yes or no and he says "No." Again, each of the answers takes progressively more deliberation but this time because the required certainty of the answers is increasing. This illustrates the tradeoff between certainty and time.

Now suppose that you ask your friend to give you any answer he can within five seconds. When asked if there are more than one hundred offices in the building, he answers "Yes." You ask if there are between one hundred and five thousand offices and he says "Probably." Finally you ask if there are between one and two thousand offices and he says that he simply doesn't know. With a fixed time limit, as each question becomes more specific, the certainty of the answer must be lowered. This is an example of the tradeoff between certainty and specificity.

2.2. Censored Production Rules

The current research examines the tradeoff between time and certainty, when specificity is kept constant. Michalski and Winston [1986] first presented a logic in which the certainty of an inference could be varied to conform to cost constraints. Variable Precision Logic (VPL) used censored production rules to encode both domain and control information. The rules take the form:

P -> D ⌊ C

read *If P then D unless C*,

where P is the premise, D is the decision, and C is the censor. The premise is a conjunction of literals; the decision is a single literal; and the censor is a disjunction of literals. These rules embody both object level and control level information. The object level information is expressed in the logical semantics and the control level information resides in the fact that censors represent exceptions. Since exceptions are by definition false most of the time, we have certain expectations concerning the character of the inferences made with such rules. These expectations may be used to control the inferences.

Two means of assigning a logical semantics, with associated procedural semantics to these rules are: the passive interpretation and the active interpretation. Winston, in his learning system [Winston, 1986], uses the passive interpretation.

This says that rules with exceptions are logically equivalent to:

$$P \mathbin{\&} \neg C \rightarrow D$$

or alternatively,

$$P \rightarrow (D \vee C) \ .$$

Winston exploits the control level knowledge in the reasoning process by allocating only one–step effort to determining the censor values and unlimited effort to checking the premises. If a censor cannot be shown to be true, it is assumed false.

In the active interpretation proposed by Michalski & Winston [1986], censored production rules represent the conjunction of the two rules:

$$P \mathbin{\&} \neg C \rightarrow D$$

$$P \mathbin{\&} C \rightarrow \neg D$$

or equivalently,

$$P \rightarrow (D \oplus C),$$

where $\oplus$ denotes the exclusive–or operator. The control level information concerning the censors is expressed by representing the strength of the inference when the censor value is known and when the censor value is unknown. The details of representing beliefs will be explained in section 3.3. Since the censors are exceptions, the inference will be quite strong even if their values are unknown. Thus in time critical situations, meaningful inferences may be made even when the resources devoted to determining the censor values are limited.

It is clear from the logical semantics that the active interpretation of a censored production rule provides more information than the passive interpretation does. The active interpretation allows conclusions concerning the negation of the decision, while the passive interpretation can only make positive conclusions. The two approaches also differ in that they make different uses of the information that exceptions are false most of the time in order to achieve efficient inference. The passive interpretation uses this information to assume that censors are false if they cannot be proven true, whereas the active interpretation uses this information to make acceptably strong inferences even without complete knowledge of the censor values.

The following example will make this discussion more concrete. Suppose that on Sundays I go for a drive unless the weather is bad. This can be expressed by the rule:

$$\text{Sunday} \rightarrow \text{Drive car} \mathrel{\lfloor} \text{Weather is bad.}$$

From this we can conclude that if it is Sunday and the weather is good, I will drive my car; also if it is Sunday and the weather is bad, I will not drive my car. If it is not Sunday then nothing can be concluded. Since the exclusive–or operator is symmetric, one might be tempted to rewrite this rule as

$$\text{Sunday} \rightarrow \text{Weather is bad} \mathrel{\lfloor} \text{Drive car.}$$

With respect to the logical semantics, this rule is correct but it is not intuitive. There seems to be an inherent asymmetry in the unless operator. This asymmetry is due to the expectation concerning the truth value of the decision and the exception. Given the premise, the decision is true most of the time, whereas the censor is false most of the time. The first rule expresses the idea that on most Sundays I drive my car and on most Sundays the weather is not bad. The second rule expresses the idea that on most Sundays the weather is bad and on most Sundays I do not drive my car.

2.3. Making Expectations Quantitative

In order to quantify the expectations inherrent in the semantics of censored production rules, Michalski & Winston [1986] introduce subjective point probabilities. A censored production rule is then written

$$P \rightarrow D \lfloor C : \gamma\, \delta,$$

where $\gamma = \text{prob}(D|P)$ and $\delta = \text{prob}(D|P\&\neg C)$. A δ value less than one represents an incomplete censor. Michalski & Winston suggest performing inference by using the δ value when the censor is known to be false and the γ value when the censor value is unknown. There are a number of problems in reasoning with this representation scheme due to inherent limitations of Bayesian inference when using point probabilities. These problems will be discussed in section 3.1. In addition, this approach leads to a discrete kind of reasoning behavior: either the censor value is known or not; thus either the δ value or the γ value is being used. Haddawy [1986] presents an implementation which displays this type of behavior. In the next section an extension of this notion is presented where the censor value can be known with varying degrees of certainty. This results in a continuous behavior along the entire spectrum between the two extremes.

2.4. The Provided Operator

If it is useful to augment production rules with exception conditions which are false most of the time, we might want also to include conditions which are true most of the time. For this purpose, the provided operator is introduced as the dual of "unless." To express the fact that P implies D provided C, we write

$$P \rightarrow D \lceil C,$$

where P is a conjunction of literals, D is a single literal, and C is a conjunction of literals. This single rule is defined as the conjunction of the two rules

$$P \,\&\, C \rightarrow D$$

$$P \,\&\, \neg C \rightarrow D$$

or equivalently,

$$P \rightarrow (D \leftrightarrow C)$$

This is semantically equivalent to writing

$$P \rightarrow D \lfloor \neg C$$

but more directly expresses the idea that we wish to convey. For example, to express the fact that if interest rates are low I will buy a new car provided I have sufficient funds we could write:

$$\text{Low-interest} \rightarrow \text{Buy-car} \lceil \text{Sufficient-funds}.$$

This could be expressed using "unless" as follows:

$$\text{Low-interest} \rightarrow \text{Buy-car} \lfloor \neg\text{Sufficient-funds}.$$

3. REPRESENTING AND REASONING WITH BELIEFS IN VPL

Along with the logical semantics presented above, the VPL system associates a probabilistic semantics with rules and facts. The basis of this semantic interpretation is the Dempster–Shafer theory of belief.

3.1. Comparison of Approximate Inference Methods

The primary goal of this thesis was to develop a system capable of reasoning with incomplete and uncertain information under time constraints. The system was to be capable of varying the certainty of its inferences to produce a result within a given time limit. To achieve this behavior, a suitable approximate inference scheme needed to be chosen and the notion of the certainty of an inference defined within that scheme. The definition of the certainty of an inference should satisfy two criteria: it should correspond to our intuitive understanding of certainty; and the certainty of any inference should increase monotonically with the time allotted. The second criterion guarantees that as more information is brought to bear on a problem, the certainty of the result increases. To choose an approximate inference scheme, three uncertainty calculi were considered:

- Mycin Certainty Factors
- Bayesian Inference
- Dempster–Shafer Theory

Shortliffe [1976] introduced the notion of certainty factors for modeling belief in the Mycin medical expert system. The certainty of a hypothesis given some evidence is expressed using two factors: the measure of belief and the measure of disbelief.

Shortliffe & Buchanan [1984] define these measures as:

MB[h,e] = x means "the measure of increased belief in the hypothesis h, based on the evidence e, is x"

MD[h,e] = y means "the measure of increased disbelief in the hypothesis h, based on the evidence e, is y"

These values are governed by the following restrictions

$$0 \leq MB[h,e] \leq 1$$

$$0 \leq MD[h,e] \leq 1$$

The certainty factor of a hypothesis h given some evidence e is then defined as

$$CF[h,e] = MB[h,e] - MD[h,e],$$

where

$$-1 \leq CF[h,e] \leq 1$$

The certainty factor indicates the net belief in a hypothesis. $CF > 0$ indicates that there is more reason to believe a hypothesis than to disbelieve it, while $CF < 0$ indicates that there is more reason to disbelieve a hypothesis than to believe it. $CF = 0$ indicates either a lack of evidence concerning a hypothesis or conflicting evidence.

Measures of belief and disbelief are combined and propagated according to the following functions [Shortliffe & Buchanan, 1984]:

Conjunction:

$$MB[h_1 \& h_2, e] = min(MB[h_1, e], MB[h_2, e])$$

$$MD[h_1 \& h_2, e] = max(MD[h_1, e], MD[h_2, e])$$

Disjunction:

$$MB[h_1 \vee h_2, e] = max(MB[h_1, e], MB[h_2, e])$$

$$MD[h_1 \vee h_2, e] = min(MD[h_1, e], MD[h_2, e])$$

Propagation:

Given a rule P $\rightarrow$ D with $MB'[D,P]$ and $MD'[D,P]$, if P is not known to be true or false but is believed with a CF between −1 and +1, based on some evidence e; then the belief in D is given by:

$$MB[D,e] = MB'[D,P] \cdot max(0, CF[P,e])$$

$$MD[D,e] = MD'[D,P] \cdot max(0, CF[P,e])$$

Given two rules $P_1 \rightarrow D$ and $P_2 \rightarrow D$, the belief and disbelief in the decision are computed according to the function:

$$MB[D, P_1 \& P_2] = 0 \qquad \text{if } MD[D, P_1 \& P_2] = 1$$

$$MB[D, P_1 \& P_2] = MB[D, P_1] + MB[D, P_2](1 - MB[D, P_1]) \qquad otherwise$$

$$MD[D, P_1 \& P_2] = 0 \qquad \text{if } MB[D, P_1 \& P_2] = 1$$

$$MD[D, P_1 \& P_2] = MD[D, P_1] + MD[D, P_2](1 - MD[D, P_1]) \qquad otherwise$$

Certainty factors worked well in Mycin's limited medical domain but have no clear theoretical basis and their applicability in other domains is dubious.

Bayesian inference is the oldest uncertainty calculus. There is an extensive body of literature on both the theory and application of Bayesian inference. Thus it was the most thoroughly investigated alternative to the chosen Dempster-Shafer theory. When applied to rule-based systems, this method interprets rules as representing conditional point probabilities and facts as simple point probabilities. For example, the rule E $\rightarrow$ H with certainty α is interpreted as prob(H|E)=α, where E is called the evidence and H the hypothesis. A model of a domain is built by assigning prior probabilities to the evidence and hypotheses. These probabilities are then updated to yield posterior probabilities when the user enters new information. Bayes theorem states that

$$prob(H \mid E_1,...,E_n) = \frac{prob(E_1,...,E_n \mid H) \cdot prob(H)}{prob(E_1,...,E_n)}$$

A theoretically correct implementation of Bayesian inference requires knowledge of the conditional probability of every fact given every combination of other facts. For example, to infer $prob(H)$ from $prob(E_2)$ and $prob(E_3)$ requires knowledge of $prob(H \mid E_2 \& E_3)$, $prob(H \mid \neg E_2 \& E_3)$, $prob(H \mid E_2 \& \neg E_3)$, and $prob(H \mid \neg E_2 \& \neg E_3)$. Thus for a knowledge base with n facts we require knowledge of 2^n conditional probabilities. This is a prohibitively large amount of information to gather and store. To solve this problem, there are two approaches suggested in the literature: the use of the maximum entropy principle and the use of simplifying assumptions with the L′ interpolation heuristic.

Maximum entropy is a means of computing the complete conditional probability distribution given only partial information concerning the conditional probabilities. Any conditional probabilities supplied along with any posterior probabilities are considered to be a set of constraints. The remaining conditional probabilities are chosen so that the constraint set is satisfied and the least additional bias is introduced. This is done by choosing the remaining conditional probabilities so that the entropy of the system is maximized [Duda, et al. 1979]. The computation is performed using the simplex method of optimization. Unfortunately, this method is too computationally costly for use as an inference method, particularly when response time is of concern.

The need for the 2^n conditional probabilities can be partially alleviated by assuming independence among the evidence events. In this case, Bayes theorem can be written as

$$prob(H \mid E_1,\ldots,E_n) = \prod_{i=1}^{n} \frac{prob(E_i \mid H)}{prob(E_i)} \cdot prob(H)$$

But by Bayes theorem again

$$prob(E_i \mid H) = \frac{prob(H \mid E_i) \cdot prob(E_i)}{prob(H)}$$

So

$$prob(H \mid E_1,\ldots,E_n) = \prod_{i=1}^{n} \frac{prob(H \mid E_i)}{prob(H)} \cdot prob(H)$$

Thus we have an expression for the probability of H given a conjunction of evidence events in terms of the probability of H given each of the events individually, as might be expressed in the form of rules in an expert system. To propagate probabilities, the posterior probability of H as a result of observing event E′ can be expressed as

$$prob(H \mid E') = prob(H \mid E_1,\ldots,E_n) \cdot prob(E_1, \ldots, E_n \mid E')$$
$$+ \; prob(H \mid \overline{E_1,\ldots,E_n}) \cdot prob(\overline{E_1, \ldots, E_n} \mid E')$$

The problem in using this is that commonly observations correspond to $prob(E_i \mid E')$ individually and not $prob(E_1, \ldots, E_n \mid E')$. The L′ interpolation heuristic is an approximate solution

* The presentation here closely follows that of Konolige in Appendix D of [Duda, et al. 1979].

to this updating problem proposed by Duda, et al. [1976]*. L′ is both continuous and monotonic but only corresponds to Bayesian theory when applied to certain evidence. The L′ heuristic is defined as

$$prob(H \mid E') = \frac{L'_A L'_B O(H)}{1+L'_A L'_B O(H)}$$

where

$$O(H) = \frac{prob(H)}{1-prob(H)}$$

$$L'_A = \frac{O(H \mid A')}{O(H)}$$

$$L'_B = \frac{O(H \mid B')}{O(H)}$$

$$prob(H \mid A') = prob(H \mid A) \cdot prob(A \mid E') + prob(H \mid \neg A) \cdot prob(\neg A \mid E')$$

Unfortunately, L′ displays extreme nonlinear behavior for certain combinations of evidence. This means that small changes in the probability of the evidence can produce large changes in the probability of the hypothesis. A change in the probability of a piece of evidence should not cause a correspondingly greater change in the probability of the hypothesis. This type of behavior would suggest that there are additional factors affecting the hypothesis. The worst cases of non-linearity occur for very high and very low certainty of evidence. If $L'_B \gg 1$, we have the behavior shown in figure 2. Almost all the effect on $prob(H \mid E')$ takes place when $prob(H \mid A')$ has reached the value $10/L'_B$.

At this point we turn to the question of how to define the notion of certainty of inference for a Bayesian inference scheme using point probabilities. Within this framework, the most appealing definition of certainty is in terms of entropy.

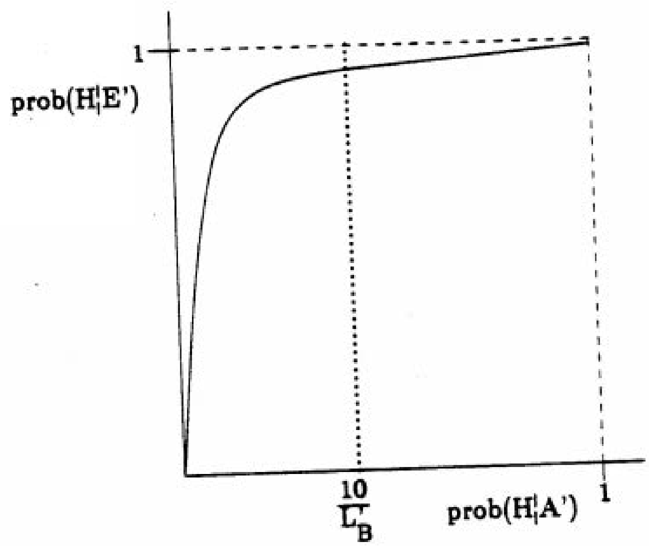


Figure 2. Non-linearity of L' interpolation for $L'_B \gg 1$.

Given a probability distribution P, the entropy is defined as

$$H = -\sum P_i \log P_i$$

where the P_i are the probabilities of the individual events. The certainty of a fact is then defined as the inverse of its entropy value. A fact known with probability 0.5 has the lowest certainty and a fact known to be absolutely true or false has the highest certainty. The problem with this formulation is that it violates the monotonicity criterion. For example, if the probability of a proposition is inferred to be 0.8 given 3 seconds inference time, there is no guarantee that the inferred probability will not be 0.5 given 5 seconds. Under this definition of certainty, an increase in inference time resulted in a decrease in certainty. It is assumed here that time is equivalent to amount of search permitted and that unknown facts are given a posterior probability equal to 0.5 or their priors. A definition of the certainty of an inference is needed which corresponds to the notion of amount of information. As will be shown below, Dempster–Shafer theory employs a simple definition of certainty which provides such a notion and satisfies both the monotonicity and intuitivity criteria mentioned above. For a more general discussion of the properties of various uncertainty calculi in the context of inference see Wise & Henrion [1984].

3.2. Dempster–Shafer Theory

Dempster–Shafer Theory (DST) is a generalization of Bayesian inference which provides a method of explicitly representing the ignorance inherrent in knowledge and of reasoning in the absence of certain information. The method was originally proposed by Dempster [1968] and later adapted to reasoning in discrete domains by Shafer [1976]. By allowing for ignorance, DST avoids the problem of requiring the complete set of conditional probabilities mentioned earlier. The central ideas of DST will now be presented. The presentation follows closely that of Garvey, et al. [1981].

The belief in a proposition A is represented by a *Shafer interval* [s(A), p(A)] which is a subinterval of the unit interval [0 .. 1]. s(A) is called the *support* and p(A) the *plausibility* of the proposition A. These values can be thought of as upper and lower bounds on the probability of A. The uncertainty of a proposition is then defined as the width of the interval: u(A) = p(A) - s(A). The *precision of the probability estimate* for the proposition A is then defined as PPE(A) = 1–u(A). The PPE of the decision is the measure which will be used to characterize the certainty of an inference. Henceforth we will refer to this as the *precision of the inference*. When the precision of the probability estimate is one for all propositions, the system reduces to a Bayesian scheme. An example will illustrate the contrast between this representation and that of point probabilities. Suppose I throw a die and ask you what the probability of a six coming up is. You would answer one sixth. Now if I tell you that the die is not necessarily fair, you would have to say that the probability is somewhere between zero and one. Reasoning schemes using point probabilities have no way of capturing this type of uncertainty, which corresponds to a lack of observations. If you observed a hundred throws of the die, you would be able to form a hypothesis concerning its fairness and thus narrow your estimate of the probability of a six coming up. As will be shown later, it is exactly this type of uncertainty that needs to be represented in the VPL system.

Dempster's rule is a method of integrating information from independent sources. In Dempster's calculus, propositions are represented as subsets of the exhaustive set of possibilities Θ, the *frame of discernment.* A *basic probability assignment* m maps elements of 2^Θ to a value in the interval [0 .. 1]. Given a proposition which represents some subset of the possibilities contained in the frame of discernment, m(A) represents the probability mass constrained to stay in A but otherwise free to move, called its *basic probability mass.* This represents our ignorance because we cannot further subdivide our belief and restrict movement of the probability mass [Barnett, 1981]. m(Θ) represents the residual uncertainty of the domain. m is defined as follows:

$$m(\phi) = 0$$

$$m(A) \in [0..1]$$

$$\sum_{A \subseteq \Theta} m(A) = 1$$

where ϕ denotes the empty set. The support for a proposition A is then defined as the total mass attributed to A and all of its subsets. This represents the total belief in A. The plausibility is one minus the support for the complement of A.

$$s(A) = \sum_{A_i \subseteq A} m(A_i)$$

$$p(A) = 1 - s(\neg A)$$

To make this discussion more concrete, consider the following example. Suppose you are thinking of moving to a new city. You have several alternatives in mind and would like to rate them according to your primary criterion, the type of women who live there. You like blondes and red heads but not brunettes. So your frame of discernment representing the exhaustive set of possibilities is $\Theta = \{$blond, brunette, red head$\}$. In order to make your decision, you decide to travel to each city and see what the proportion of blonds and red heads to brunettes is. You visit the first city and walk through the streets with your note pad, recording the hair color of each woman you see. But you chose to visit the city in the winter and some of the women are wearing hats so you do not know what their hair color is; it could be any one of the three. You see 40 blonds, 20 red heads, 30 brunettes, and 10 women with hats, making a total of 100 women. Based on this, the basic probability assignment m is defined as

$$\text{m(blond)} = 0.4$$

$$\text{m(red head)} = 0.2$$

$$\text{m(brunette)} = 0.3$$

$$\text{m(hat)} = \text{m(blond} \vee \text{red head} \vee \text{brunette)} = \text{m}(\Theta) = 0.1$$

The support for the proposition A $\equiv$ blond$\vee$red head is then

$$\text{s(A)} = \text{m(blond)} + \text{m(red head)} = 0.6$$

and the support for $\neg$A $\equiv$ brunette is

$$\text{s}(\neg\text{A}) = \text{m(brunette)} = 0.3.$$

The plausibility of A is p(A) = 1-s(¬A) = 0.7. So the prob(A) ∈ [0.6 .. 0.7]. The uncertainty in the domain is just m(Θ) = 0.1. Dempster–Shafer theory allows us to represent ignorance by assigning a probability mass to a statement which contains several alternatives, in this case m(hat). In this way we can account for a lack of observations and uncertain observations, both of which represent forms of incomplete information.

Two basic probability assignments m_1 and m_2 which provide evidence concerning a common frame of discernment are combined using *Dempster's orthogonal sum rule* to yield a new basic probability assignment. The combined effects of m_1 and m_2 are represented by a unit square. Suppose we have a frame of discernment Θ = {a,b,c} and the propositions A ≡ a∨b and ¬A ≡ c. Figure 3 shows the unit square representing the combination of two basic probability mass functions m_1 and m_2. The belief committed exactly to the combination A∩Θ is then the area of the rectangles corresponding to the intersection, i.e., $m_1(A) \cdot m_2(\Theta) + m_1(\Theta) \cdot m_2(A)$. The total mass allocated to a given subset C of Θ is

$$\sum_{A_i \cap B_j = C} m_1(A_i) m_2(B_j).$$

Referring to the above example, the total mass allocated to A is

$$m(A) = m_1(A) \cdot m_2(A) + m_1(A) \cdot m_2(\theta) + m_1(\theta) \cdot m_2(A)$$

Since the combination of m_1 and m_2 must again be a basic probability assignment, the following must hold.

$$\sum_{A_i \cap B_j \neq \phi} m_1(A_i) m_2(B_j) + \sum_{A_i \cap B_j = \phi} m_1(A_i) m_2(B_j) = 1$$

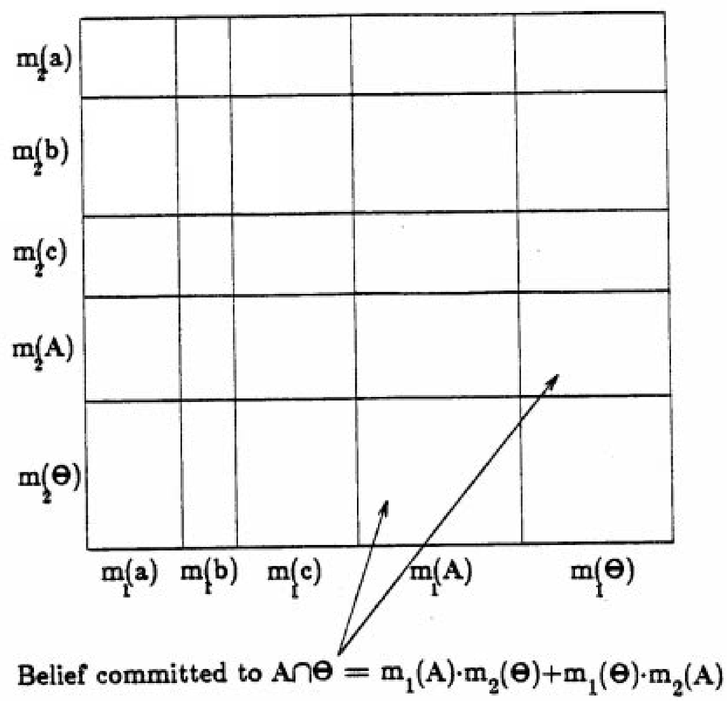


Figure 3. Dempster's orthogonal sum.

Thus Dempster's orthogonal sum rule for the new probability function for all subsets C of Θ is

$$m(C) = \frac{\sum_{A_i \cap B_j = C} m_1(A_i)m_2(B_j)}{1 - \sum_{A_i \cap B_j = \phi} m_1(A_i)m_2(B_j)}.$$

The denominator is a normalizing factor which removes the probability assigned to the empty set.

Now consider the case in which Θ contains only two values A and ¬A. The support and plausibility values are then:

$$s(A) = m(A)$$

$$p(A) = 1 - m(\neg A) = m(A) + m(\Theta).$$

Substituting these into Dempster's rule and simplifying, we obtain:

$$s(A) = 1 - \frac{(1-s_1(A))(1-s_2(A))}{1-[s_1(A)(1-p_2(A))+(1-p_1(A))s_2(A)]}$$

Dempster's rule can be expressed in terms of Shafer intervals as:

$$[a \;..\; b] \oplus [c \;..\; d] = \left[1 - \frac{\bar{a}\,\bar{c}}{1-(a\bar{d}+\bar{b}c)} \;..\; \frac{bd}{1-(a\bar{d}+\bar{b}c)}\right]$$

where ⊕ denotes the orthogonal sum of the two intervals [a .. b] and [c .. d]. This is similar to the formula given by Ginsberg [1984]. The formula is both associative and commutative so it can be applied pairwise in any order.

3.3. Dempster–Shafer Interpretation of VPL

The VPL system represents knowledge in terms of rules and facts. To apply DST to VPL, we must provide an interpretation of rules and facts in terms of Shafer intervals. The belief in a fact is represented by associating a Shafer interval with each fact. Shafer intervals are viewed as representing upper and lower bounds on the probability of a fact; thus $A_{[s,p]} \equiv prob(A) \in [s \;..\; p]$, where A is a proposition. A censored production rule should then imply information concerning the Shafer interval of its decision given the Shafer intervals of its premises and censors. Implication is interpreted as expressing conditional probability.

Four belief values are associated with each censored production rule:

$$P \rightarrow D \lfloor C : \delta^{+}, \delta^{-}, \gamma^{+}, \gamma^{-}$$

The δ^{+} value is the lower bound on the probability of D given P and ¬C; the δ^{-} value is the lower bound on the probability of ¬D given P and C; the γ^{+} value is the lower bound on the probability of D given P; and the γ^{-} value is the lower bound on the probability of ¬D given P. This is summarized below:

$$\text{prob}(D|P \ \& \ \neg C) \in [\delta^{+} \ .. \ 1]$$

$$\text{prob}(\neg D|P \ \& \ C) \in [\delta^{-} \ .. \ 1]$$

$$\text{prob}(D|P) \in [\gamma^{+} \ .. \ 1]$$

$$\text{prob}(\neg D|P) \in [\gamma^{-} \ .. \ 1]$$

The γ^{+} and γ^{-} values are constrained by the restriction that $\gamma^{+} + \gamma^{-} \leq 1$, thus

$$\text{prob}(D|P) \in [\gamma^{+} \ .. \ (1 - \gamma^{-})].$$

For example, the following rule might be used to express the fact that I read the paper before going to work unless I oversleep, which occurs once or twice a week:

Weekday-morning -> Read-paper ⌊ Oversleep : 0.9, 1, 0.6, 0.2

where the belief factors are interpreted as follows:

- the "0.9" states that on weekday mornings when I do not oversleep I read the paper at least 0.9 of the time because there are other factors which would keep me from reading the paper, such as the paper boy throwing it on the roof, which are not being considered;

- the "1" states that on weekday mornings when I oversleep I certainly do not read the paper;
- the "0.6" states that I read the paper at least three out of five weekday mornings (because I oversleep at most twice a week);
- the "0.2" states that I do not read the paper at least one out of five weekday mornings (because I oversleep at least once a week).

This formalism also allows representation of incomplete and inconsistent censors. Figure 4 shows the set theoretic interpretation of a censored production rule. Most of the premise intersects with the decision, indicating that the inference P → D is quite strong. Most of that part of the premise which is outside of the decision is covered by the censor. This censor is both incomplete and inconsistent. The censor part of a rule is incomplete if it does not cover all possible exceptions. It is inconsistent if it has a nonempty intersection with the decision. The degree of incompleteness is expressed by the amount by which $\delta^{*} < 1$ and the

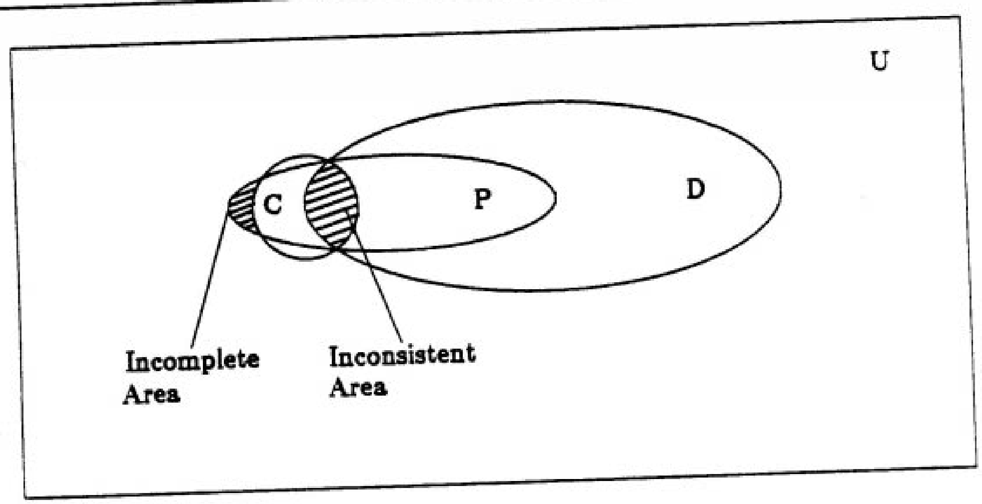


Figure 4. Set theoretic interpretation of P → D ⌊ C.

degree of inconsistency by the amount by which $\delta^- < 1$.

The *provided* condition, the dual of *unless*, is represented as:

$$P \rightarrow D \lceil C : \delta^+, \delta^-, \gamma^+, \gamma^-$$

where

$$prob(D|P \,\&\, C) \in [\delta^+ .. 1]$$

$$prob(\neg D|P \,\&\, \neg C) \in [\delta^- .. 1]$$

$$prob(D|P) \in [\gamma^+ .. 1]$$

$$prob(\neg D|P) \in [\gamma^- .. 1].$$

The set theoretic interpretation of the provided operator is shown in figure 5. Since the censor part of a provided rule is true most of the time, it is shown to cover a large part of the

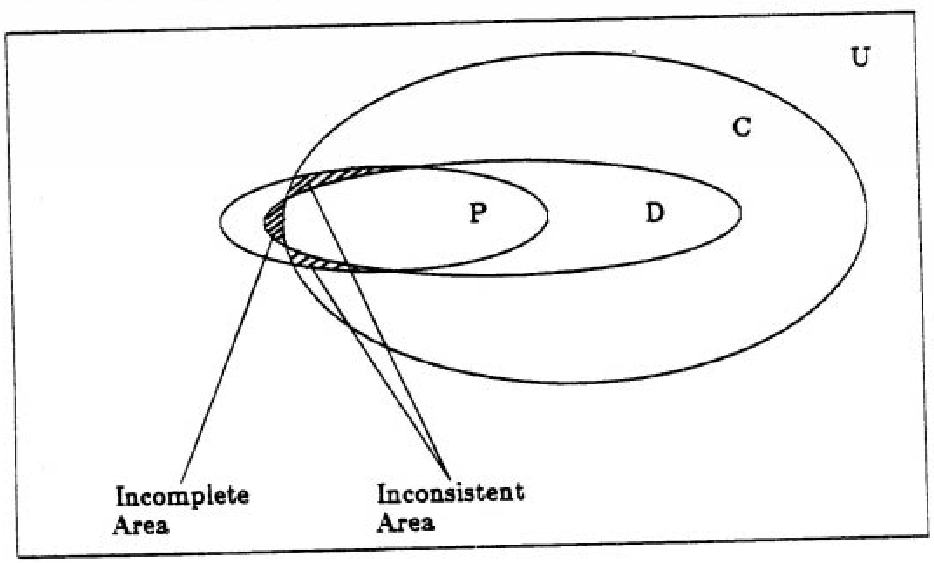


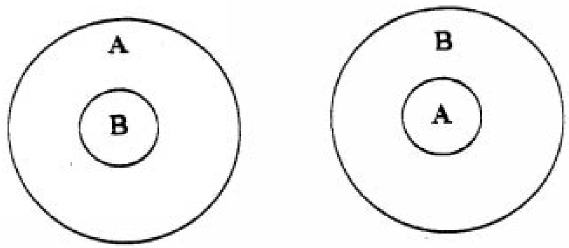
Figure 5. Set theoretic interpretation of $P \rightarrow D \lceil C$.

universe of discourse. Most of the premise intersects with the decision, indicating again that the inference P → D is quite strong. Most of the intersection of the premise with the decision is covered by the censor. The area of intersection between the premise and decision not covered by the censor represents the degree of incompleteness of the censor. The area of the premise covered by the censor but outside the decision represents the degree of inconsistency of the censor.

The above discussion has presented a representation of belief for propositional logic, but the VPL system uses a predicate logic representation. In this representation, terms containing only ground instances are equivalent to propositional logic and thus present no additional problems. However, a semantics for expressions with free variables is needed. Rules of the form A(x,y) → B(x), with an associated belief [s p] are interpreted as $\forall x,y$ prob(B(x)|A(x,y)) $\in$ [s .. p]. This is essentially a shorthand for listing rules over the entire domain of x and y. Similarly, a fact A(x) with belief [s p] is interpreted as $\forall x$ prob(A(x)) $\in$ [s .. p].

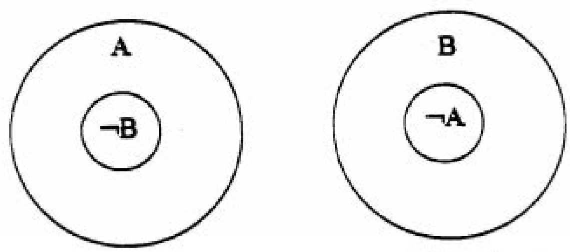
3.4. Combination & Propagation of Belief in VPL

The VPL system reasons by combining belief values via logical operators and propagating belief values across rules. This implies that a probabilistic semantics must be assigned to the combination operators &, $\vee$, and $\neg$ and to the propagation operator →. There are several ways of making this assignment, corresponding to different assumptions and interpretations of the operators. As stated earlier, Shafer intervals are interpreted as upper and lower bounds on probabilities. Within a probabilistic framework, three assumptions which can be made concerning the correlation between events are: positive correlation, negative correlation, and independence. Given two propositions A & B, they are positively correlated



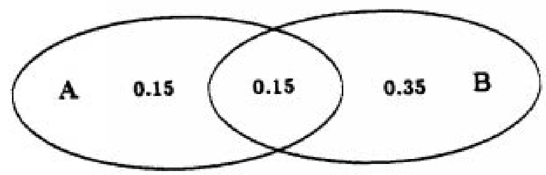
prob(A|B)=1   prob(B|A)=1

Figure 6. Positive correlation.



prob(A|¬B)=1   prob(B|¬A)=1

Figure 7. Negative correlation.



prob(A|B)=prob(A)
prob(B|A)=prob(B)

Figure 8. Independent events.

if prob(A|B) = 1 or prob(B|A) = 1, as shown in figure 6. As can be seen from the diagrams, the probability of the conjunction of the two propositions is just the minumum of the two and the probability of the disjunction is just the maximum of the two. Two propositions are negatively correlated if prob(A|¬B) = 1 or prob(B|¬A) =1, as shown in figure 7. Two propositions are independent if prob(A|B)=prob(A) and prob(B|A)=prob(B), in other words, the event of one has no influence on the event of the other. Since by definition prob(A|B) = prob(A&B)/prob(B), under independence prob(A&B) = prob(A)·prob(B). This is shown in figure 8. To summarize, under these assumptions, the combination functions for conjunction and disjunction are:

Independence: prob(A&B) = prob(A)·prob(B)
prob(A∨B) = prob(A)+prob(B)−prob(A)·prob(B).

Positive: prob(A&B) = Min(prob(A), prob(B))
prob(A∨B) = Max(prob(A), prob(B))

$$\text{Negative: } \text{prob}(A\&B) = \text{Max}(0, \text{prob}(A)+\text{prob}(B) - 1)$$
$$\text{prob}(A \vee B) = \text{Min}(1, \text{prob}(A)+\text{prob}(B))$$

The negation operator has the same definition under all these assumptions, namely prob($\neg$A) = 1−prob(A). Wise & Henrion [1984] have shown that of the three assumptions, inference under the independence assumption corresponds most closely to inference using the maximum entropy principle. Since maximum entropy can be shown to cause the least biasing of inference behavior, the independence assumption was adopted in the VPL system. When computing the conjunction or disjunction of two Shafer intervals, the result should yield upper and lower bounds on the probability of the conjunction or disjunction. Under the independence assumption, the upper and lower bounds are obtained by applying the combination functions to the support and plausibility values separately as follows:

If $\text{prob}(A) \in [s(A) .. p(A)]$

and

$\text{prob}(B) \in [s(B) .. p(B)]$

then $\text{prob}(A \,\&\, B) \in [s(A) \cdot s(B) .. p(A) \cdot p(B)]$

and

$$\text{prob}(A \vee B) \in [s(A)+s(B)-s(A) \cdot s(B) .. p(A)+p(B)-p(A) \cdot p(B)]$$

To define negation recall that $s(\neg A) = 1-p(A)$ from which it follows that $p(\neg A) = 1-s(A)$. This yields the definition:

$$\text{prob}(\neg A) \in [1-p(A) .. 1-s(A)]$$

Rules in the VPL system are interpreted as representing bounds on conditional probabilities. Standard probability theory provides a simple formula for propagating probabilities across such rules. Given a rule $P \rightarrow D$, the probability of D may be expressed as:

$$prob(D) = prob(D|P)\cdot prob(P) + prob(D|\neg P)\cdot prob(\neg P).$$

A propagation function for Shafer intervals may be derived by performing a sensitivity analysis on this formula [Dubois & Prade, 1985]. Suppose the following:

$$prob(P) \in [s(P) .. p(P)]$$

$$prob(\neg P) \in [1-p(P) .. 1-s(P)]$$

$$prob(D|P) \in [s(P \rightarrow D) .. p(P \rightarrow D)]$$

$$prob(\neg D|P) \in [s(P \rightarrow \neg D) .. p(P \rightarrow \neg D)]$$

$prob(D|\neg P) \in [0 .. 1]$, i.e., it is unknown.

Values from these bounds must be chosen which will minimize and maximize the prob(D). These values must obey the restriction that if the upper bound on prob(P) is chosen then the lower bound on $prob(\neg P)$ is used and vice versa since $prob(\neg P) = 1 - prob(P)$. To derive the lower bound, first take $prob(D|\neg P) = 0$. Then since $prob(\neg P)$ will have no effect, take $prob(\neg P) = 1-s(P)$. For prob(D|P) we are free to take $s(P \rightarrow D)$. The lower bound is then simply the product $s(P \rightarrow D)\cdot s(P)$.

To determine the upper bound, first take $prob(D|\neg P) = 1$. Since $prob(D|P) < 1$, the maximum value is yielded by taking the lower bound on $prob(\neg P)$ and the upper bound on prob(P). Thus the upper bound is given by $1-s(P)+p(P \rightarrow D)\cdot s(P)$. Since $p(P \rightarrow D) = 1-s(P \rightarrow \neg D)$, this can be simplified to $1-s(P)\cdot s(P \rightarrow \neg D)$. To summarize, the final propagation function is:

$$prob(D) \in [s(P)\cdot s(P \rightarrow D) .. 1-s(P)\cdot s(P \rightarrow \neg D)].$$

When a VPL type rule has no censors, this is the propagation function used. For the rule $P \rightarrow D : \gamma^+, \gamma^-$ the resulting expressions for the support and plausibility of the decision are:

$$s(D) = s(P) \cdot \gamma^+$$
$$p(D) = 1 - s(P) \cdot \gamma^-$$

For rules with censors, both the δ and γ values must be taken into account. We concentrate first on propagation using the δ values. Consider first δ^+ which represents the lower bound on the prob(D|P&¬C). The support for D may be obtained by taking the product of δ^+ and the lower bound on prob(P&¬C). Thus

$$s(D) = s(P)[1-p(C)] \cdot \delta^+$$

Next, consider δ^- which represents the lower bound on prob(D|P&C). The support for ¬D may be computed by taking the product of δ^- and the lower bound on prob(P&C). The plausibility of D can then be derived using the identity $p(D) = 1 - s(\neg D)$ to yield:

$$p(D) = 1 - s(P) \cdot s(C) \cdot \delta^-$$

These support and plausibility values represent the constraints which the censor value puts on the range of values which the decision may assume. The interval represented by these values is called the ***possible range*** of the decision and denoted $[s_{poss}\ p_{poss}]$. For example, if the Shafer interval for P is [1 1] then the range of the decision is equal to the range of the censor. The propagation functions for the possible range are then:

$$s_{poss}(D) = s_{poss}(P)[1-p_{poss}(C)]\delta^+$$
$$p_{poss}(D) = 1 - s_{poss}(P) \cdot s_{poss}(C) \cdot \delta^-,$$

where the references to the possible ranges on the right hand side allow the values to be propagated from rule to rule.

The possible range can leave some uncertainty concerning the value of the decision. The amount of uncertainty is equal to the width of the probability interval. This uncertainty represents the probability mass which could not be assigned exactly to D or ¬D. In other words, it is free to move between the two values. The γ values can be used to apportion this remaining probability mass. The γ values represent the probability of D given P when the censor value is unknown. When the censor value is completely unknown, they are used to distribute the entire probability mass. When the censor value is partially known, they can be used to distribute that portion of the probability mass which the censor value has failed to constrain. The range produced by use of the γ values is called the *most likely range* and is denoted $[s_{ml}\ p_{ml}]$. The corresponding propagation functions are:

$$s_{ml}(D) = s_{poss}(D) + [s_{ml}(P)\cdot\gamma^{+}][p_{ml}(C) - s_{ml}(C)]$$
$$p_{ml}(D) = p_{poss}(D) - [s_{ml}(P)\cdot\gamma^{-}][p_{ml}(C) - s_{ml}(C)].$$

Note that the propogation functions for rules without censors is just the special case of these where $s_{poss}(D) = 0$, $p_{poss}(D) = 1$, and $[p_{ml}(C) - s_{ml}(C)] = 1$. The most likely ranges of P & C appear on the right hand side to allow the most likely range to be propagated from rule to rule. For any observed fact, the possible range and most likely range are equal. Figure 9 shows some possible and most likely ranges for various values of the premise and censor.

The first four rows of the table show that both propagation functions are faithful to the logical interpretation of censored production rules. The middle three rows show the effect that varying the censor value has on the most likely and possible ranges for a premise known

P		C		D			
s	p	s	p	s_{poss}	p_{poss}	s_{ml}	p_{ml}
0	0	0	0	0	1	0	1
0	0	1	1	0	1	0	1
1	1	0	0	1	1	1	1
1	1	1	1	0	0	0	0
1	1	0.00	1.00	0.00	1.00	0.70	0.90
1	1	0.40	0.60	0.40	0.60	0.54	0.58
1	1	0.60	0.60	0.40	0.40	0.40	0.40
0.80	0.90	0.00	1.00	0.00	1.00	0.56	0.92
0.80	0.90	0.40	0.60	0.32	0.68	0.43	0.66
0.80	0.90	0.60	0.60	0.32	0.52	0.32	0.52

Figure 9. Possible and most likely ranges of the decision for the rule P→ D ⌊ C : 1, 1, 0.7, 0.1

with absolute certainty. When the censor value corresponds to a point probability, the most likely range is equal to the possible range. When the censor value is completely unknown, the possible range is [0 1] and the most likely range is a function of the γ values. When the censor has some certainty between zero and one, the most likely range is a subrange of the possible range. The possible range can be thought of as indicating the extent to which the most likely range could change given more information concerning the censor. The last three rows of the table are the same as the above three except that the premise is only known with partial certainty. Again when the censor value is a point probability, the possible and most likely ranges are equal because the uncertainty of the decision is not due to any uncertainty in the censor.

The reasoning behavior displayed by these propagation functions is a generalization of default reasoning. Default logic [Reiter, 1980] is a formalism for reasoning with incomplete information in a logical framework. Lack of information is handled by the use of assumed truth values when the truth value of a fact is unknown. Thus either the truth value of a fact is known or it is a complete assumption. VPL, on the other hand, allows partial information to be incorporated into the default reasoning process, resulting in conclusions with varying amount of default character between the two extremes. This is reflected in the most likely range of the decision as a function of the PPE of the censor value. When the censor value is known as a point probability, the most likely range of the decision is strictly a function of the δ values and the evidence. When the censor value is unknown, the most likely range is strictly a function of the γ values and the evidence. When the censor value is partially known, the most likely range is a function of the δ values, the γ values, and the evidence, weighted toward the δ or γ values according to the PPE of the censor. The δ values can be thought of as the belief in the decision when all of the available evidence is considered, and the γ values can be thought of as the default belief in the decision when information concerning the censors is limited. Figure 10 shows the characteristic curves of the propagation function for various values of the PPE of the censor for the rule:

$$\mathrm{P} \rightarrow \mathrm{D} \lfloor \mathrm{C} : 1, 1, 0.8, 0.2$$

Note that as the PPE of the censor increases, the midpoint of the probability interval for the decision approaches the default value of 0.8.

To relate the combination and propagation functions presented above back to censored production rules, consider the following two rules:

$$P_1 \,\&\, P_1' \rightarrow D \lfloor C_1 \vee C_1' : \delta^*_1, \delta_1, \gamma^*_1, \gamma_1$$

$$P_2 \,\&\, P_2' \rightarrow D \lfloor C_2 \vee C_2' : \delta^*_2, \delta_2, \gamma^*_2, \gamma_2.$$

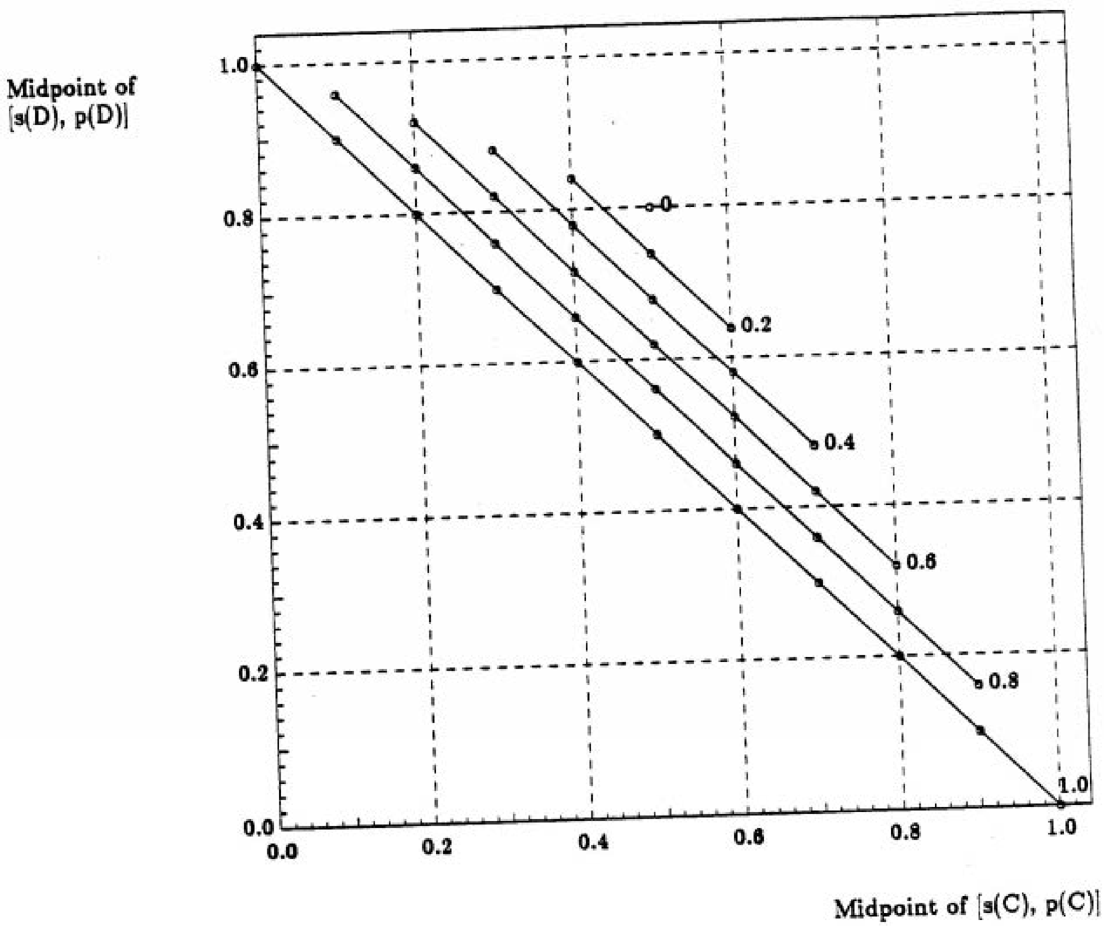


Figure 10. Characteristic curves for the rule
P → D ⌊ C : 1, 1, 0.8, 0.2
for PPE(C) = 0, 0.2, 0.4, 0.6, 0.8, 1.

Given the belief in the premises and censors, we wish to know the belief in the decision D. For each rule, the beliefs in the premises are combined using the combination function for conjunction; the beliefs in the censors are combined using the combination function for disjunction. Using these values and the propagation functions, the belief that each rule attributes to D is calculated separately, yielding two most likely and two possible ranges. The most likely ranges and possible ranges are finally combined using Dempster's orthogonal sum to arrive at the total belief in D as a function of all available evidence.

3.4.1. Tweety Example

The following example illustrates the way in which the VPL system reasons with uncertain information. Consider the rule base shown in figure 11, based on an example from Michalski & Winston [1986]. These rules indicate whether a bird can fly based on evidence concerning its condition and the type of bird it is. The first rule states that a bird can fly unless it is a special bird or is in an unusual condition. The next three rules define what an unusual condition is and the last five rules enumerate the different kinds of special birds.

(is-bird x) => (flies x)
⌊ (is-special-bird x) ∨ (is-in-unusual-condition x) : 1, 1, 0.8, 0.1

(is-dead x) => (is-in-unusual-condition x) : 1, 0
(is-sick x) => (is-in-unusual-condition x) : 1, 0
(has-broken-wing x) => (is-in-unusual-condition x) : 1, 0

(is-penguin x) => (is-special-bird x) : 1, 0
(is-ostrich x) => (is-special-bird x) : 1, 0
(is-emu x) => (is-special-bird x) : 1, 0
(is-kiwi x) => (is-special-bird x) : 1, 0
(is-domestic-turkey x) => (is-special-bird x) : 1, 0

where x is universally quantified.

Figure 11. Tweety example rule base.

Now we would like to give the system various observations about some bird named Tweety and ask it if Tweety can fly. We first tell the system that Tweety is a dead bird:

(is–bird Tweety) : 1 1
(is–dead Tweety) : 1 1

When asked if Tweety can fly the system responds

Most Likely Range = [0.00 0.00]
Possible Range = [0.00 0.00],

indicating that Tweety certainly cannot fly. The certain premise combined with the certain censor yield the negation of the decision. Now, if the certainty in Tweety's death is reduced to

(is–dead Tweety) : 0.7 0.8,

the resulting belief in his ability to fly is

Most Likely Range = [0.24 0.27]
Possible Range = [0.00 0.30].

The possible range indicates that there is no support for Tweety's ability to fly and strong support against it. The most likely estimate of his ability to fly lies toward the upper end of the possible range since most birds can fly. Based on the most likely range, we can conclude that Tweety probably cannot fly. Next we add some suspicion that Tweety is in fact a kiwi:

(is–kiwi Tweety) : 0.3 0.5.

This combines with the belief in his death to further reduce the belief that he can fly:

Most Likely Range = [0.17 0.19]
Possible Range = [0.00 0.21].

Note that the additional evidence also causes the PPE to increase. Suppose now that we tell the system that Tweety is healthy and normal:

(is–in–unusual–condition Tweety) : 0 0
(is–special–bird tweety) : 0 0

It is now certain that he can fly:

Most Likely Range = [1.00 1.00]
Possible Range = [1.00 1.00].

3.5. Time Constrained Inference

With the framework of the propagation functions presented above, inference is quite straight–forward. Using rules to represent the effects of evidence on a hypothesis reduces the problem of inference to that of search through the space of rules and facts. Suppose we wish to determine the belief in a fact entailed by a knowledge base of rules and facts. Call this fact the query. If the knowledge base contains a fact which unifies with the query, this value is returned. Otherwise, all the rules whose decision parts unify with the query are sought. For each of these rules' premises and censors we search for unifying facts in the knowledge base and if none are found, we search for rules. This search process continues in a recursive manner. Since all evidence linked through some rules to the query can effect its belief, the search must be exhaustive. This search process can be described as backward chaining exhaustive search. When the search is complete, the truth values are propagated from the evidence through the rules back to the original fact.

We have just described the way in which a complete search through the knowledge base is performed. If there is insufficient time for a complete search, the VPL system limits the depth of search used to determine censor values. This is done by searching in a breadth first manner for rules which imply the censors until the time limit is reached. The amount of search preformed to determine the censor values is called the *censor chaining depth* and is defined to be the length of the deepest sub-path which has a censor as its root. The censor itself is considered to be at depth one. Figure 12 shows the points at which search is

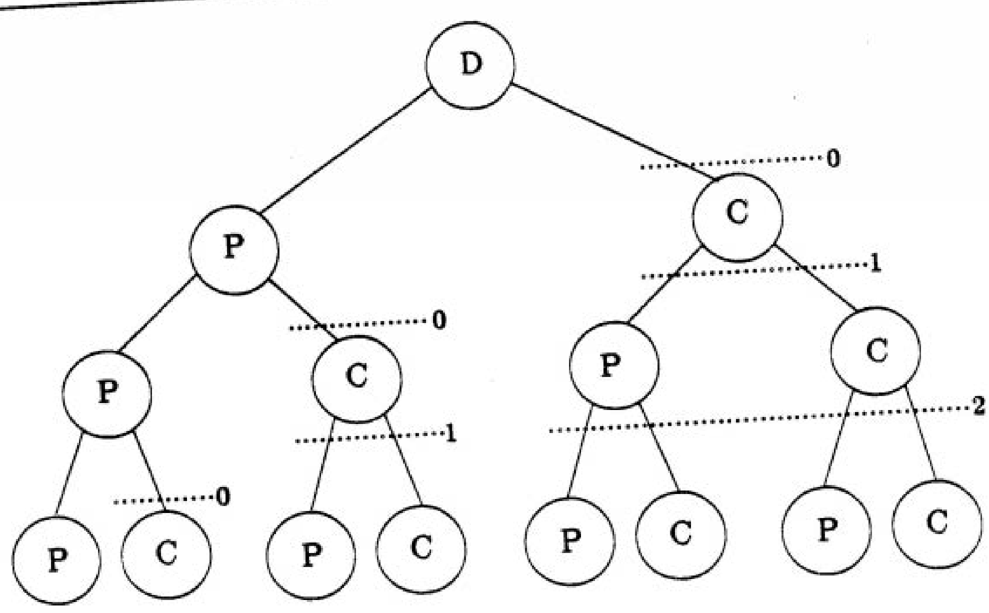


Figure 12. Censor chaining depths of 0, 1, 2.

terminated for censor chaining depths of 0, 1, and 2 through a space of seven rules. A censor chaining depth of zero examines no censors at all. A depth of one examines only the first level censors but does not search for rules implying their values. A depth of two searches for rules one level below these censors. When search is prematurely terminated at some censor chaining depth, there will be several nodes (premises and censors) whose truth values are unknown. These nodes are assigned values of [0 1]. These unknown values along with all the belief values found are then propagated back. As can be seen from figure 12, the greater the censor chaining depth, the more information is potentially found. Of course there is no guarantee that more information will be found with more search since the values of all the terminal nodes can be [0 1]. However, the point is that the amount of information found monotonically increases with search depth. An increase in search will never be accompanied by a decrease in the amount of information found but the amount of information can remain constant if the search is looking in places where no information exists.

With the search process clearly described, it is now possible to show that the Dempster–Shafer notion of uncertainty satisfies the monotonicity criterion.

Theorem

The precision of VPL type inference increases monotonically as a function of inference time or censor chaining depth.

Proof

Since the PPE of a proposition is defined as the inverse of its uncertainty, it suffices to show that the uncertainty decreases monotonically as a function of inference time or censor chaining depth. As described above, when search is terminated early, the belief [0 1] is assigned to nodes on the frontier of the search tree. Thus we need to show that substituting the value [0 1] for the belief of a fact results in a greater uncertainty than if the true value of the fact were used. This must be shown for each of the combination and propagation functions.

CONJUNCTION:

By definition, prob(A & B) $\in$ [s(A)·s(B) .. p(A)·p(B)]

If prob(B) $\in$ [0 .. 1] then prob(A & B) $\in$ [0 .. p(A)].

Since s(B) $\geq$ 0 and p(B) $\leq$ 1,

s(A)·s(B) $\geq$ 0 and p(A)·p(B) $\leq$ p(A).

Thus p(A)·p(B) – s(A)·s(B) $\leq$ p(A).

So the uncertainty is greatest when the Shafer interval for B is [0 1].

DISJUNCTION:

By definition,

prob(A$\vee$B) $\in$ [s(A)+s(B)–s(A)·s(B) .. p(A)+p(B)–p(A)·p(B)]

If prob(B) $\in$ [0 .. 1] then prob(A $\vee$ B) $\in$ [s(A) .. 1].

First we show that s(A)+s(B)–s(A)·s(B) $\geq$ s(A).

Since s(A) $<=$ 1, s(B) $\geq$ s(A)·s(B).

So s(B)–s(A)·s(B) $\geq$ 0.

from which it follows that

s(A)+s(B)–s(A)·s(B) $\geq$ s(A).

Next we show that p(A)+p(B)–p(A)·p(B) $\leq$ 1.

Since 1–p(A) $\geq$ 0 and 1–p(B) $\geq$ 0,

(1–p(A))(1–p(B)) $\geq$ 0.

So 1 – (1–p(A))(1–p(B)) $\leq$ 1.

Combining these two results we have

p(A)+p(B)–p(A)·p(B) – s(A)+s(B)–s(A)·s(B) $\leq$ 1–s(A).

DEMPSTER'S ORTHOGONAL SUM:

First note that

[a .. b] $\oplus$ [0 .. 1] = [a .. b].

So we need to show that

[a .. b] $\oplus$ [c .. d] = [e .. f]

implies that

(f – e) $\leq$ (b – a).

Recall that if m is a basic probability assignment and Θ is the frame of discernment then m(Θ) is the uncertainty of the domain. Thus the above problem can be stated as showing that

if m = m1 $\oplus$ m2

then m(Θ) $\leq$ m1(Θ).

According to the definition of Dempster's orthogonal sum,

$$m(\Theta) = \frac{\sum_{A_i \cap B_j = \Theta} m_1(A_i)m_2(B_j)}{1 - \sum_{A_i \cap B_j = \emptyset} m_1(A_i)m_2(B_j)}$$

The denominator simplifies to 1, yielding:

$$m(\Theta) = m_1(\Theta)m_2(\Theta)$$

Since for any m, $m(\Theta) \leq 1$,

$$m(\Theta) \leq m_1(\Theta).$$

PROPAGATION:

Recall that the general form of the propagation function for the rule P → D with associated belief [s(P → D) .. p(P → D)] is

$$\text{prob}(D) \in [s(P) \cdot s(P \rightarrow D) \; .. \; 1 - s(P) \cdot s(P \rightarrow \neg D)].$$

If the Shafer interval for P is [0 1] then prob(D) ∈ [0 .. 1]. Since any Shafer interval is a subinterval of the unit interval, the uncertainty of D is greatest when the belief in P is [0 1]. This completes the proof.

In addition to the monotonic increase in the precision with increasing inference time, there are certain cases in which the certainty of the decision itself actually can be shown to increase with increasing inference time. As mentioned earlier, the closer a decision is to 0.5, the less certain it is and the closer it is to 0 or 1, the more certain it is. Consider the situation in which there is only one rule per hypothesis and all the evidence is either true or false. Suppose we have the rule

P → D ⌊ C : 1 1 0.7 0.3

and the assertion P : 1 1. If C is true and we perform censor chaining, then D is inferred to be false. If C is false and we perform censor chaining then D is inferred to be true. If we do not chain on C then the value of D is inferred to be 0.7. Clearly 0.7 is closer to 0.5 than 0 or 1. This decrease in certainty which accompanies a decrease in censor chaining depth results because the γ^* value is always less than the δ^* value and since C represents an exception, γ^* is never less than 0.5. So $0.5 < \gamma^* < 1$, from which it follows that the certainty decreases with decreasing censor chaining depth or increasing inference time.

3.5.1. Autonomous Car Example

A simple example will serve to illustrate the way in which the VPL system exploits the tradeoff between inference time and precision. Suppose it is a Sunday afternoon and your fully autonomous car is taking you for a drive. Suddenly a boulder rolls out into the road ahead of you. Your car has only a fraction of a second to decide what to do.

As an initial reflex action, the car applies the brakes lightly. If the car is not traveling too fast, detects the obstacle far in advance, and the brakes work as expected, then application of the brakes will stop it in time to avoid a collision. Of course, this is not always the case so the car must decide on possible alternative courses of action. One such alternative is to turn off the road to avoid the obstacle. In order to do this, the car must not wait too long

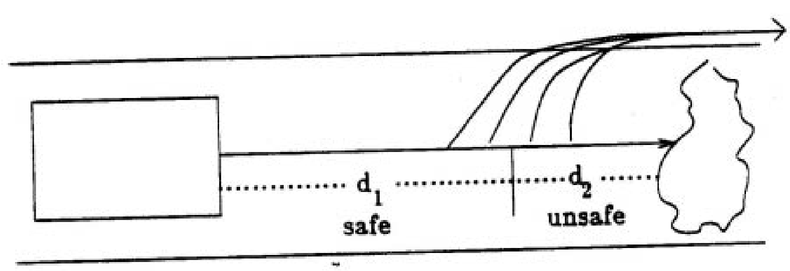


Figure 13. Safe turning distances from obstacle.

Rules:

R1 *(obstacle-ahead) (~ speed-distance-ratio high)*
=> (use-brakes)
⌊ (tire-traction poor) (brake-condition poor) : 1, 1, 0.7, 0.2

R2 *(obstacle-ahead) (~ on-bridge)*
=> (turn-off-road)
⌊ (ditch-on-side-of-road) : 0.95, 0.9, 0.8, 0.2

R3 *(on ice road)*
=> (tire-traction poor)
⌊ (using-chains) : 1.0, 0.8, 0.9, 0.05

R4 *(on gravel road) => (tire-traction poor) : 0.85, 0.1*

R5 *(tire-type snow) => (tire-traction good) : 0.9, 0*

R6 *(tire-traction good) => (~ tire-traction poor) : 1, 0*

R7 *(temperature below-freezing) (road-appearance shiny)*
=> (on ice road) : 0.9, 0.1

R8 *(construction-site) (sound-of-pebbles-hitting-underside-of-car)*
=> (on gravel road) : 0.9, 0.1

Facts:

(obstacle-ahead) : 1 1
(speed-distance-ratio high) : 0.05 0.15
(temperature below-freezing) : 0.1 0.2
(road-appearance shiny) : 0 0
(construction-site) : 1 1
(sound-of-pebbles-hitting-underside-of-car) : 0.1 0.3
(brake-condition poor) : 0.1 0.2
(using-chains) : 0 0
(tire-type snow) : 1 1
(on-bridge) : 0 0
(ditch-on-side-of-road) : 1 1

Figure 14. Autonomous car knowledge base.

((USE-BRAKES) (0.168 : 3) (0.164 : 2) (0.078 : 1) (0.026 : 0))
((TURN-OFF-ROAD) (0.023 : 1) (0.015 : 0))
((OBSTACLE-AHEAD) (0.002 : 0))
((ON-BRIDGE) (0.002 : 0))
((DITCH-ON-SIDE-OF-ROAD) (0.002 : 0))
((SPEED-DISTANCE-RATIO HIGH) (0.002 : 0))
((SPEED-DISTANCE-RATIO MEDIUM) (0.002 : 0))
((SPEED-DISTANCE-RATIO LOW) (0.002 : 0))
((TIRE-TRACTION POOR) (0.120 : 1) (0.116 : 0))
((TIRE-TRACTION FAIR) (0.011 : 0))
((TIRE-TRACTION GOOD) (0.021 : 0))
((BRAKE-CONDITION POOR) (0.002 : 0))
((BRAKE-CONDITION FAIR) (0.002 : 0))
((BRAKE-CONDITION GOOD) (0.002 : 0))
((SQUEAKY-BRAKES) (0.002 : 0))
((ON ICE GROUND) (0.008 : 0))
((ON ICE ROAD) (0.017 : 0))
((ON GRAVEL GROUND) (0.008 : 0))
((ON GRAVEL ROAD) (0.016 : 0))
((TEMPERATURE ABOVE-FREEZING) (0.002 : 0))
((TEMPERATURE BELOW-FREEZING) (0.002 : 0))
((ROAD-APPEARANCE ROUGH) (0.002 : 0))
((ROAD-APPEARANCE SHINY) (0.002 : 0))
((CONSTRUCTION-SITE) (0.002 : 0))
((SOUND-OF-PEBBLES-HITTING-UNDERSIDE-OF-CAR) (0.002 : 0))
((USING-CHAINS) (0.002 : 0))
((TIRE-TYPE SNOW) (0.002 : 0))
((TIRE-TYPE STICKY) (0.002 : 0))
((TIRE-TYPE REGULAR) (0.002 : 0))

Figure 15. Data base of inference times for autonomous car.

otherwise it will either not clear the obstacle or it will have to turn so sharply that it will spin. The situation is depicted in figure 13. The car must decide if it is going to turn before it has reached a distance d_2 from the obstacle.

Figure 14 shows a hypothetical rule base which such an intelligent car might contain. The first two rules express conditions under which a maneuver can be successfully completed without causing damage to the car. Rule R1 says that the car can use its brakes to avoid a collision if the speed to distance ratio is not too high unless either the road or brakes are in poor condition. Rule R2 says that the car can turn off the road to avoid a collision if it is not on a bridge unless there is a ditch on the roadside. Rules R3 through R8 are for determining the road and brake conditions. Rules R4 through R8 have no censors and thus have only two belief factors, the γ values. Following the rules are the facts known to the system. To demonstrate the precision/time tradeoff, the VPL system is given three progressively increasing time limits to decide which course of action to take. As the time limit is increased, more information is found and thus the precision of the inference increases. The first inference simply returns the default values, while the last inference finds all relevant evidence. Interaction with the system has been translated into English to simplify the presentation.

The inference performed by the autonomous car in order to decide on the best course of action occurs at two conceptual levels: the meta level and the object level. First inference at the meta level is performed to determine how much time will be devoted to determining the merits of each alternative course of action. To do this the car determines the total amount of time it has for the inference by measuring the distance d_1 shown in figure 13 and its current rate of travel, taking into consideration that it is currently applying the brakes. It then looks in a data base of inference times in order to decide how to apportion this time. The data base contains guaranteed inference times for various uniform censor chaining depths for all possible queries in the system. This information can either be gathered through the rule base analyzer described in section 4.3 or through experience. In this example, the available time is apportioned to allow all alternatives an equal censor chaining depth. Other strategies such as devoting more time to alternatives considered to be more important could also be used. Once the available time has been allotted, the merits of each alternative are decided at the object level, using the meta level generated time constraints.

ENTER Command or make query of system
> Decide whether to use brakes in 0.03 second.

Uniform censor chaining depth of 0 achieved in 0.024 seconds elapsed time.

RESULT: Most Likely Range = [0.60 0.83]
Possible Range = [0.00 1.00]

ENTER Command or make query of system
> Decide whether to turn off road in 0.02 second.

Uniform censor chaining depth of 0 achieved in 0.015 seconds elapsed time.

RESULT: Most Likely Range = [0.80 0.80]
Possible Range = [0.00 1.00]

Figure 16. Deciding among alternatives within 0.05 second.

Suppose first that the car is traveling at 90 miles per hour and has only 0.05 second to decide which course of action to take. Examination of the data base of inference times in figure 15 shows that within the available time, a censor chaining depth of zero can be achieved for deciding both the merits of using the brakes and turning off the road. Rather than specifying a chaining depth of zero for each inference, the inference system is given a time limit. This allows the system to use any small additional time over that required for uniform search to depth zero to perform some additional search. In other words, the system may be able to perform an amount of search between uniform depth zero and uniform depth one. Specification of a time limit on the inference as opposed to a fixed censor chaining depth allows maximum flexibility.

The interaction with the VPL system is shown in figure 16. The system is given 0.03 second to decide if the brakes should be used. The inference is completed in 0.024 second, achieving a uniform censor chaining depth of 0 and possibly a greater non-uniform chaining depth. The possible range of [0 1] indicates that the inference is purely based on default information. The most likely range is 0.60 to 0.83.

The system is given 0.02 second to decide whether the car should turn off the road and finishes the inference in 0.015 second. The probability that turning off the road will succeed lies in the interval 0 to 1, the most likely range being the point value 0.8. Based on this information, the car decides to turn off the road in order to avoid the obstacle.

ENTER Command or make query of system
> Decide whether to use brakes in 0.1 second.

Uniform censor chaining depth of 1 achieved in
0.072 seconds elapsed time.

RESULT: Most Likely Range = [0.54 0.76]
Possible Range = [0.00 0.91]

ENTER Command or make query of system
> Decide whether to turn off road in 0.025 second.

Uniform censor chaining depth of 1 achieved in
0.023 seconds elapsed time.

RESULT: Most Likely Range = [0.00 0.10]
Possible Range = [0.00 0.10]

Figure 17. Deciding among alternatives within 0.125 second.

Next, suppose the car is traveling at 35 miles per hour. This time is has a total of 0.125 second to make a decision. Examination of the data base of inference times shows that within this amount of time, a uniform censor chaining depth of one can be achieved for both alternatives. Figure 17 shows the interaction with the VPL system. Allocating 0.1 second to considering use of the brakes allows some information pertaining to the condition of the road and the brake condition to be taken into consideration. This results in a possible range of 0.00 to 0.91 and a most likely range of 0.54 to 0.76. The additional information brought to bear on the problem increases the precision of the inference.

The system is given a time limit of 0.025 second to decide on the merits of turning off the road to avoid the collision. This is sufficient time for the system to find that there is a ditch on the side of the road and thus decide that the probability of completing this maneuver without damaging itself is in the range 0 to 0.10, with a most likely range of 0 to 0.10. The most likely range is equal to the possible range because there is no uncertainty in the censor value. The uncertainty of the decision is purely a function of the δ values of the rule. Based on the most likely range, the best course of action is for the car to use its brakes. The possible range indicates, however, that this decision could be reversed in the light of additional evidence.

Finally, suppose that the car is traveling at safe rate of 20 miles per hour. This allows 0.225 seconds for deliberation. The maximum censor chaining depth for deciding whether to turn off the road is one so the same amount of time is allocated to this alternative as in the previous scenario. The remaining time is sufficient time for a complete search to determine whether to use the brakes as well. As the interaction depicted in figure 18 shows, the increased evidence brought to bear on evaluating the likely effectiveness of using the brakes

ENTER Command or make query of system
> Decide whether to use brakes in 0.2 second.

Uniform censor chaining depth of 3 achieved in
0.174 seconds elapsed time.

RESULT: Most Likely Range = [0.71 0.88]
Possible Range = [0.61 0.91]

ENTER Command or make query of system
> Decide whether to turn off road in 0.025 second.

Uniform censor chaining depth of 1 achieved in
0.023 seconds elapsed time.

RESULT: Most Likely Range = [0.00 0.10]
Possible Range = [0.00 0.10]

Figure 18. Deciding among alternatives within 0.225 second.

increases the precision of the inference significantly. It is now certain that the best course of action is to use the brakes. Using this information, the car applies the brakes and comes to a safe halt.

The results of this example show that the VPL system was able to make a rational deci-

The results of this example show that the VPL system was able to make a rational deci-

The results of this example show that the VPL system was able to make a rational decision about the best course of action to avoid a collision even though both time and information were limited. The effect of the time limit on the precision of the inference is transparent to the user, as reflected in the probability range of the answer. The "possible range" shows the effect of the immediate evidence on the conclusion and indicates to the user whether the decision could possibly be reversed given more information. The "most likely range" augments the immediate evidence with past experience to produce an answer of varying default

4. IMPLEMENTATION

The VPL system is implemented in Common Lisp and runs on a Symbolics 3640 lisp machine. The system consists of six main modules, comprising a total of 1800 lines of code. The modules are: the parser, the data base, the inference engine, the unifier, the rule base analyzer, and the user interface.

4.1. Typed Logic Representation

Knowledge in the VPL system is represented using a typed logic formalism. Type information specifies the finite domain of each predicate argument. This is done via type and predicate declarations. Before a predicate is used, both it and the types of its arguments must be declared. This information is used in two ways: to insure that information supplied by the user is semantically meaningful, and to constrain the search space. The parser performs various forms of type checking when rules and facts are entered. Constant predicate arguments must be of the correct type and the argument domains of a variable common to several predicates in a rule must have a nonempty intersection. For example, the rule shown in figure 19 would produce an error because the intersection of types t1, t2, and t3 is empty. Types are considered to intersect if their domains share common symbols.

The type information is used to constrain search by incorporating it into the unification procedure. Conceptually, when two variables are unified, the intersection of their types is formed. Two terms unify only if their bindings are compatible and their types have a non-empty intersection. As the search progresses, the types of variables become increasingly constrained. At some points, the types become empty, allowing those branches to be pruned from the search tree. Typed unification is implemented by generating all ground instances

Types:
t1 = {a b}
t2 = {b c}
t3 = {a c}

Predicates:
(P t1)
(Q t2)
(R t3)

(P x) & (Q x) => (R x) : 1, 0

Figure 19. Inconsistent variable types.

consistent with a particular binding when a rule is unified with a query. The type and binding information is expressed in this enumeration. A ground instance of a term unifies with a rule only if its arguments are elements of the argument types of the decision. Notice that the rules are stored in a compact form and only expanded when used.

4.2. Inference Engine

The inference engine is the system component in which search, combination, and propagation of beliefs is performed. There is a clear distinction in the implementation between the search phase of inference and the belief propagation phase. Inference is initiated by the user entering a query as shown in the autonomous car example earlier. The query can have either a censor chaining depth limit or a time limit associated with it. If the user specifies a censor chaining depth, the system chains to that depth and stops as shown in figure 12. If a time limit is specified, the system allocates 2/3 of this time to search and 1/3 to propagation. This proportion was derived empirically and produces actual inference times which are a few hundredths of a second below the limit given. For time constrained search, the system has available a data base of minimum inference times for each possible query. These are the times for a censor chaining depth of zero. If the time limit specified is below this limit, a message stating that the time limit is insufficient even for minimal search is issued.

```
(defun infer (query state)

 (setf state (initialize-state state query))

; Search all pure premises
   (do ((currterm (car (state-Pq state)) (car (state-Pq state))))
         ((null (state-Pq state)))
     (pop (state-Pq state))
     (setf state (evaluate-subgoal currterm state 'ante)))

; Search along the censor chains
   (when (and (cadr (state-Cq state))
              (< (get-elapsed-time) (* 0.6 *time-limit*))
              (> *depth* 0))
; Search down level by level
     (do ((depth-left *depth* (1- depth-left)))
         ((or (null (cadr (state-Cq state)))
              (>= (get-elapsed-time) (* 0.6 *time-limit*))
              (zerop depth-left)))
; Flip censor chain queues
        (setf (state-Cq state) (nreverse (state-Cq state)))

        (do ((currterm (caar (state-Cq state)) (caar (state-Cq state))))
            ((or (and (null (car (state-Cq state)))
                      (setf *depth-searched* (1+ *depth-searched*)))
                 (>= (get-elapsed-time) (* 0.6 *time-limit*))))

          (pop (car (state-Cq state)))
          (setf state (evaluate-subgoal currterm state 'censor)))))

   (cons (perform-calculations state) state))
```

Figure 20. Prioritised search algorithm.

The search strategy is backward chaining, prioritized breadth-first search. The algorithm is shown in figure 20. The algorithm works with three main data structures: the premise queue, the censor queue, and the stack of operation requests. These are all part of the state structure. First the query is put on the premise queue through a call to "initialize-state." The queue is then processed as follows. The first element on the queue is taken off and processed by function "evaluate-subgoal." The premises of any rules matching this subgoal are added to the queue. The search proceeds until this queue is empty. Censors encountered during this search are added to the censor queue. All queue additions occur in routine "update-for-new-rules" called from routine "evaluate-subgoal." Next, if time and censor chaining depth allow, the subgoals on the censor queue are processed. The censor queue actually consists of two queues. Subgoals are added to the second queue and taken off the first. When the first queue is empty, the queues are flipped. In this way the system keeps track of the censor chaining depth. Censor chaining proceeds down the search tree level by level until either the queue is exhausted or the time limit or the censor chaining depth are reached.

Whenever a conjunction of premises, a disjunction of censors, or a rule is processed, a request for the appropriate combination or propagation function is put on the stack of operations requests. The stack is processed in the propagation phase. Facts found along the search are put on a value-list for access in the propagation phase. If at any time along the search, the time limit or censor chaining depth are reached, the search is terminated. For all remaining subgoals on the censor queue, the value [0 1] is put on the value-list.

The final step in the search algorithm is to call the propagation routine "perform-calculations." This routine processes each request on the stack of operations requests. The results of each calculation are put on the value-list. The result of the last request on the stack is the answer to the original query. Additional features of the inference engine such as user query templates are described in the user manual, Appendix 7.1.

4.3. Rule Base Analyzer

The use of a typed logic representation means that the VPL system knows of all legal queries which a user could make. Based on the predicate and type declarations the system can enumerate these possible queries. This knowledge is exploited in the rule base analyzer, which measures the inference times for queries using various censor chaining depths. The analyzer can be run in two modes: minimum analysis and full analysis. In the first mode, only the inference times for a censor chaining depth of zero are measured. These times are stored in a data base of inference times and used to guarantee that the user specified time limit will at least allow minimum search. In full analysis mode, inference times for all chaining depths from zero to the maximum for each query are measured. This feature allows a knowledge engineer to produce an inference time profile for a rule base. Such a profile gives him a feel for the amount of information which can be gathered in a given amount of time for any given query. Full analysis of a rule base is a very time consuming process but it is only done during rule base development and thus is not a frequently performed computation.

5. EXPERIMENTATION

The effectiveness of the VPL system was tested in two ways. First a series of controlled experiments using automatically generated knowledge bases was run to test the effect of various knowledge base characteristics on system behavior. To test the applicability of the VPL system to "real world" problems, the ESTIMATOR system for estimating construction project costs was developed using the VPL system as the inference engine.

5.1. Controlled Experiments

The implemented VPL system exploits the tradeoff between the precision of the probability estimate and inference time by varying the amount of search performed on rule censors. The precision of any VPL type inference is purely a function of the beliefs available to the system and those found using the amount of search allowed by the time limit. The information found by a search strategy is a function of the way in which a search proceeds through a knowledge base which is itself largely a function of the rule base structure. An interesting question is then how the quality of the precision/time tradeoff is affected by the precision of the rules and assertions and the rule base structure. To test the effects of various manifestations of these two characteristics, the Rule Base Generator (RBG) was written and a series of controlled experiments using different knowledge base configurations was run. The RBG generates a knowledge base described by the following set of parameters.

- Belief value of positive assertions
- Belief value of negative assertions
- Belief factors for the rules

- Number of premises per rule
- Number of censors per rule

- Number of premises to chain on
- Number of censors to chain on

- Number of rules per premise
- Number of rules per censor

- Number of premise rules to chain on
- Number of censor rules to chain on

The knowledge bases contain assertions which are characterized as either positive or negative. For example, a positive assertion is prob(A) ∈ [0.8 .. 1] and a negative assertion prob(A) ∈ [0 .. 0.2]. The belief values of these two types can be specified. The number of premises and censors to chain on specifies the number of premises and censors in each rule which have rules implying their value. The number of rules per premise and censor indicates how many of these rules there are. The number of premise and censor rules to chain on specifies how many of these rules will in turn have rules implying the values of their premises and censors. These eleven parameters allow generation of an extremely diverse set of knowledge bases.

Using the RBG, five different rule base structures were generated. The rule bases all used rules with the beliefs: $\delta^{+} = 1$, $\delta^{-} = 1$, $\gamma^{+} = 0.8$, $\gamma^{-} = 0$. For each of these structures, four different knowledge bases, corresponding to different sets of assertions were generated. There were two values of positive assertions and two values of negative assertions, corresponding to higher and lower PPE's, used in all four combinations. One of the knowledge bases for Rule Base 1 is shown in appendix 7.2.1. The five rule base configurations are shown in figures 21 through 25, while figures 26 through 30 show the precision versus time tradeoff for each rule base structure and the set of four belief value combinations. Note that the graphs in figures 26 through 30 demonstrate experimentally the monotonicity result which was proved earlier.

Rule Base 1 is configured with one premise and censor per rule, two rules per premise and censor, and only one of these rules chained on. Note that the top level decision is treated as a premise. The precision versus time tradeoff for the set of four belief value combinations is shown in figure 26. The four graphs represent the PPE of the top level decision for increasing inference times, i.e., increasing depths of censor chaining. The leftmost point represents a censor chaining depth of zero and the rightmost point a censor chaining depth of five.

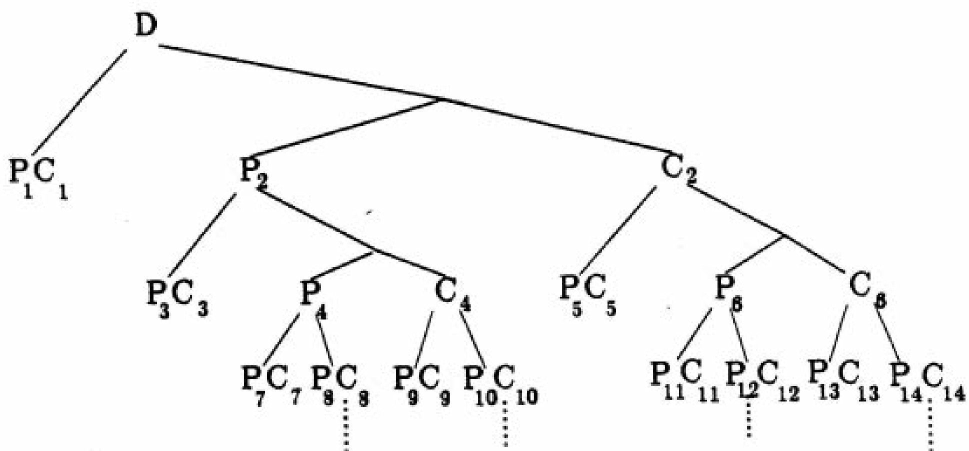
As the graphs for Rule Bases 1,2,3, and 5 show, the precision/time tradeoff follows a law of diminishing returns. In Rule Base 1, the PPE increases sharply up to an inference time of about 8 seconds and then tapers off. In the worst case for Rule Base 1, i.e., knowledge base c, the inference time can be reduced from the maximum of 25 seconds down to 8 seconds by only sacrificing a degree of the PPE of about 0.09. Thus we realize a savings of 68% of the inference time required for complete search and only pay with 9% decrease in the PPE. For Rule Bases 2, 3, and 5 the worst case tradeoffs are 67% vs 6%, 85% vs 5%, and 69% vs 7% respectively.

Rule Base 4, on the other hand, shows alternating periods of increasing PPE and little change. This behavior is most pronounced i.e. has the greatest difference between periods of increasing PPE and periods of no change, for knowledge base a, which has the most precise assertions. The effect is least pronounced for knowledge base d, which has the least precise assertions. Thus for this rule base structure, the behavior is highly dependent on the precision of the information. Comparison of the rule base structures shows that Rule Base 4 is the only one in which the censor has only one rule implying its value. This structural property causes significant information to be found only at every other censor chaining depth. When information is less precise, this additional information has less of an effect than when highly precise information is found.

A comparison of the four curves for each of the other rule bases shows that the general shape remains the same for the various PPE's of the assertions. The precision of the assertions simply shifts the curves up or down. Whereas in Rule Bases 1,4, and 5, the curves are more or less evenly distributed along the y axis, in Rule Bases 2 and 3, the curves clearly group into sets. System behavior with Rule Base 2 seems to be sensitive to the precision of positive assertions. With Rule Base 3, the system is sensitive to the combined PPE's of the assertions. When both types of assertions have a high degree of precision, the curve is high on the y axis; when one or the other is imprecise, the curves are in the middle of the scale; and when both types are imprecise, the curve is low on the scale.

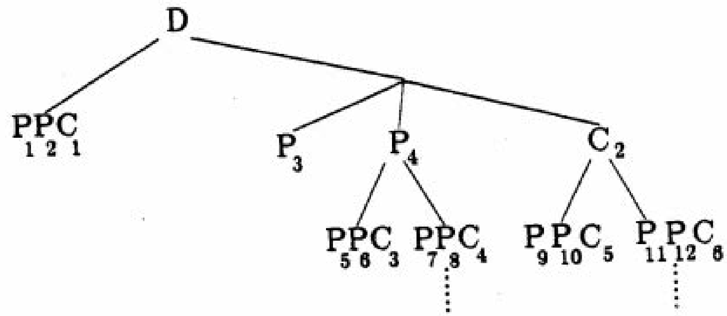
In addition, while the precision of inference in Rule Bases 1,2, and 3 is more sensitive to the precision of positive assertions, the precision of inference in Rule Bases 4 and 5 is more sensitive to the precision of the negative assertions. A comparison of the rule base structures shows that Rule Bases 1 through 3 have an a number of premises per rule equal to or greater than the number of censors per rule, while Rule Bases 4 and 5 have less premises than censors.

The results of these experiments show that rule base structure has a greater effect on the quality of the precision versus time tradeoff than the precision of the information, although there is some interaction between the two. In addition, for many rule bases the tradeoff follows a law of diminishing returns, allowing large savings in inference time at little cost in terms of precision. This occurs because the assertions found at the upper levels of the search tree have more influence on the decision than those lower down. Intuitively, the deeper the assertions, the more peripheral the evidence they represent. This diminishing returns type behavior is dependent on the presence of information at all levels of the search. If all the assertions in the search tree occur at the deepest level, the PPE will not increase until this level is reached, at which point it will jump to the maximum.



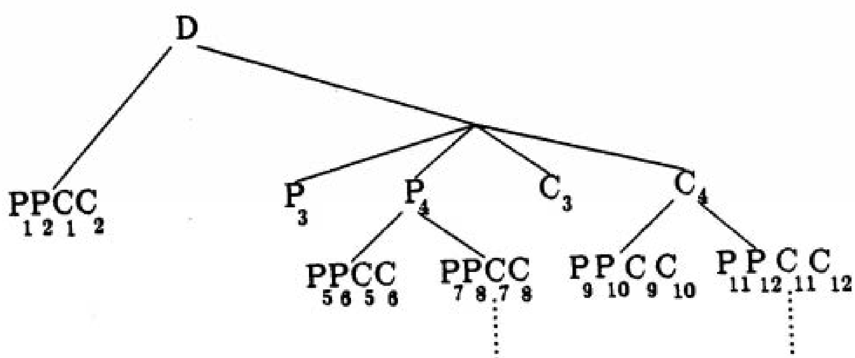
$P_1 \rightarrow D \lfloor C_1$
$P_2 \rightarrow D \lfloor C_2$

Figure 21. Structure of Rule Base 1.



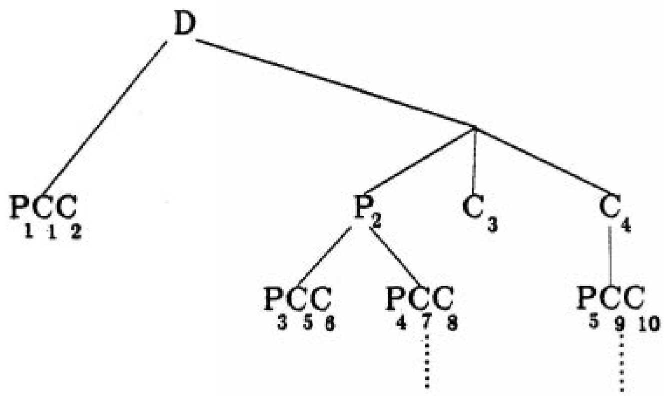
$P_1 \& P_2 \rightarrow D \lfloor C_1$
$P_3 \& P_4 \rightarrow D \lfloor C_2$

Figure 22. Structure of Rule Base 2.



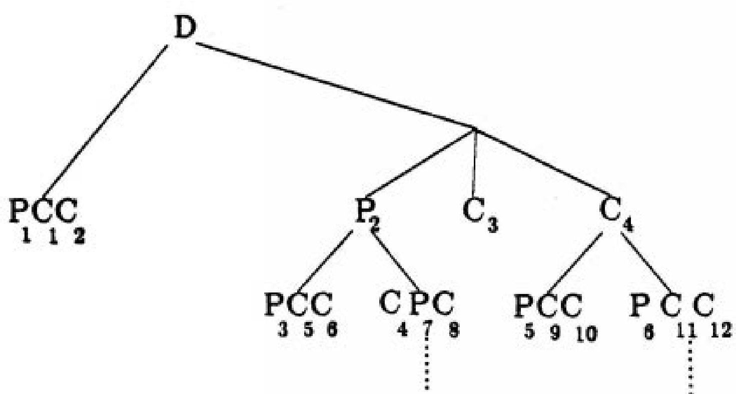
$P_1 \& P_2 \rightarrow D \lfloor C_1 \vee C_2$
$P_3 \& P_4 \rightarrow D \lfloor C_3 \vee C_4$

Figure 23. Structure of Rule Base 3.



$P_1 \rightarrow D \lfloor C_1 \vee C_2$
$P_2 \rightarrow D \lfloor C_2 \vee C_3$

Figure 24. Structure of Rule Base 4.



$P_1 \rightarrow D \lfloor C_1 \vee C_2$
$P_1 \rightarrow D \lfloor C_1 \vee C_2$

Figure 25. Structure of Rule Base 5.

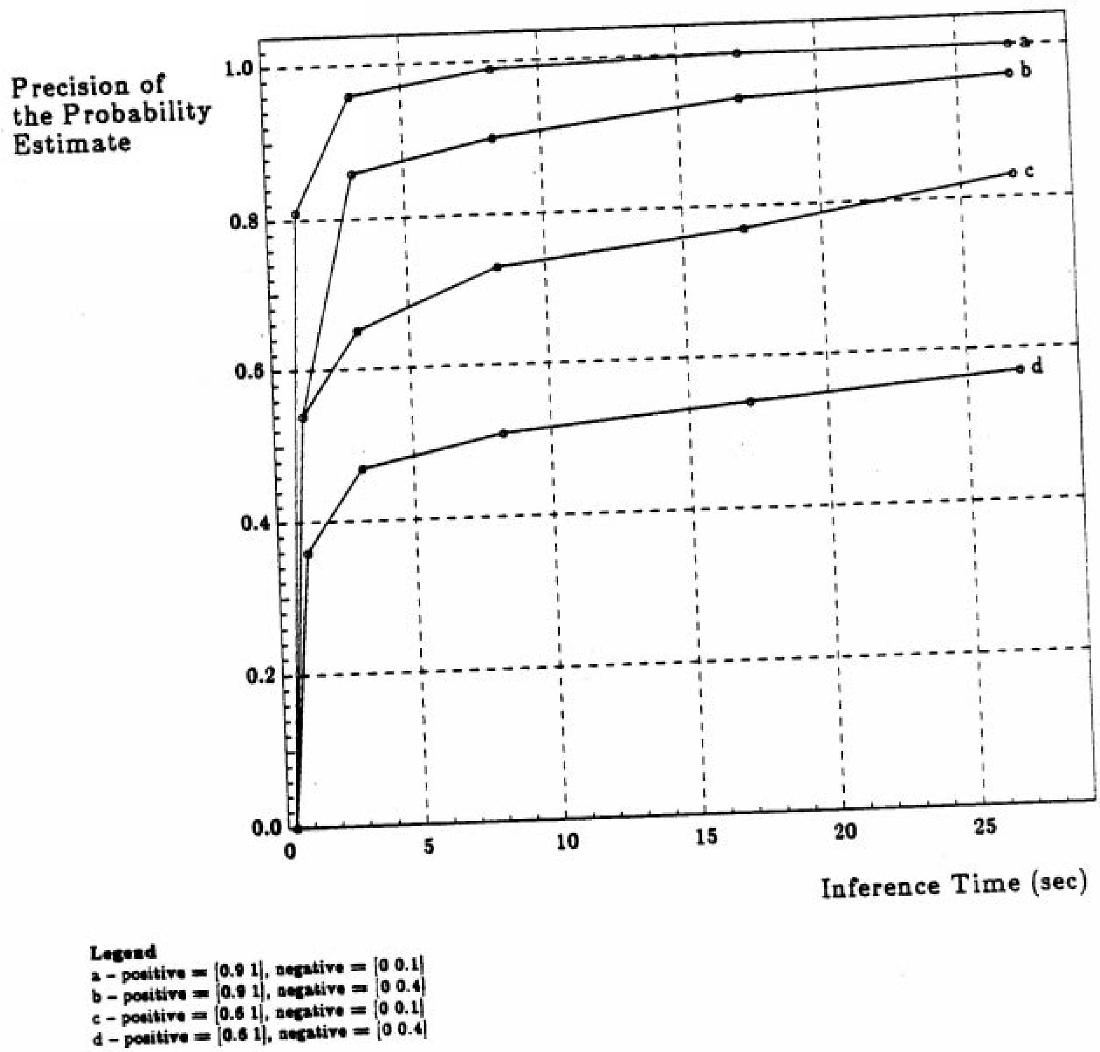
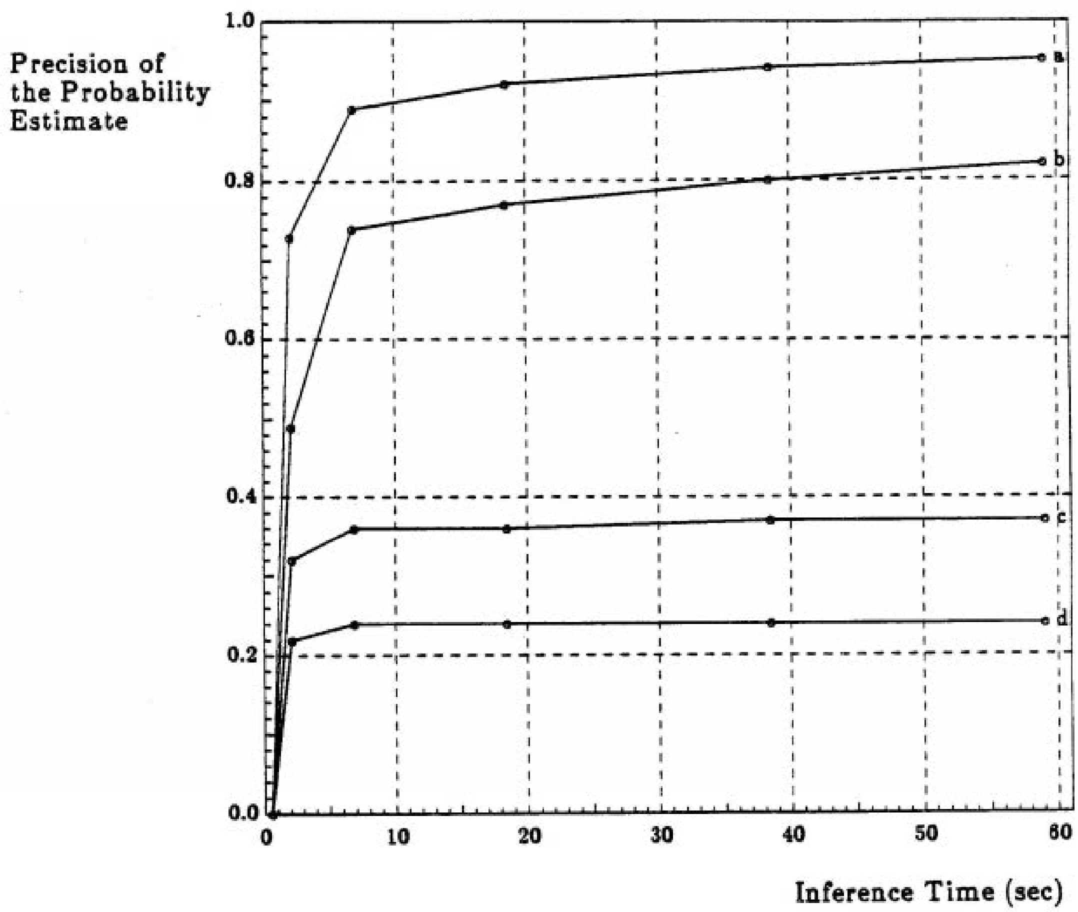


Figure 26. PPE(D) vs time for Rule Base 1.



Legend
a - positive = [0.9 1], negative = [0 0.1]
b - positive = [0.9 1], negative = [0 0.4]
c - positive = [0.6 1], negative = [0 0.1]
d - positive = [0.6 1], negative = [0 0.4]

Figure 27. PPE(D) vs time for Rule Base 2.

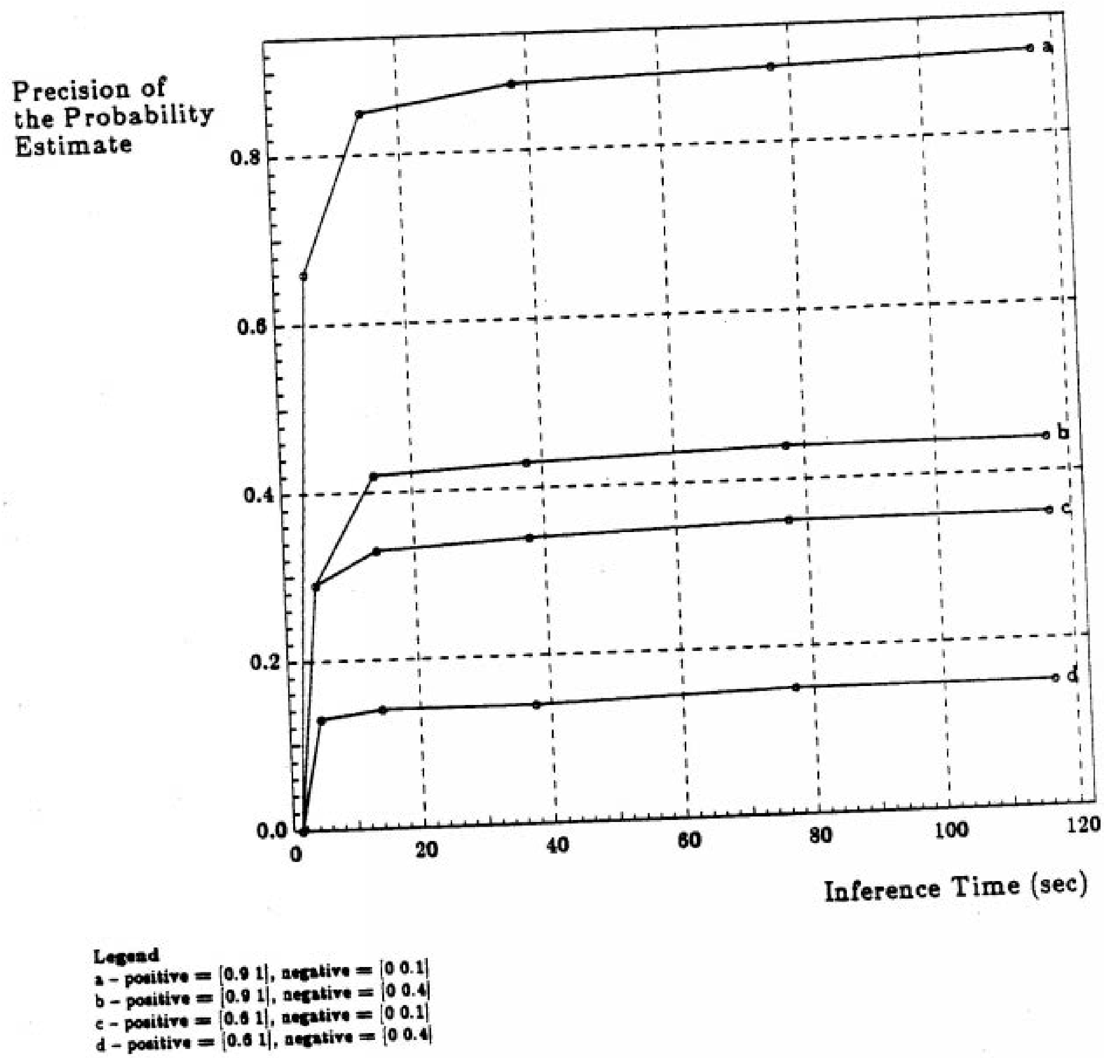


Figure 28. PPE(D) vs time for Rule Base 3.

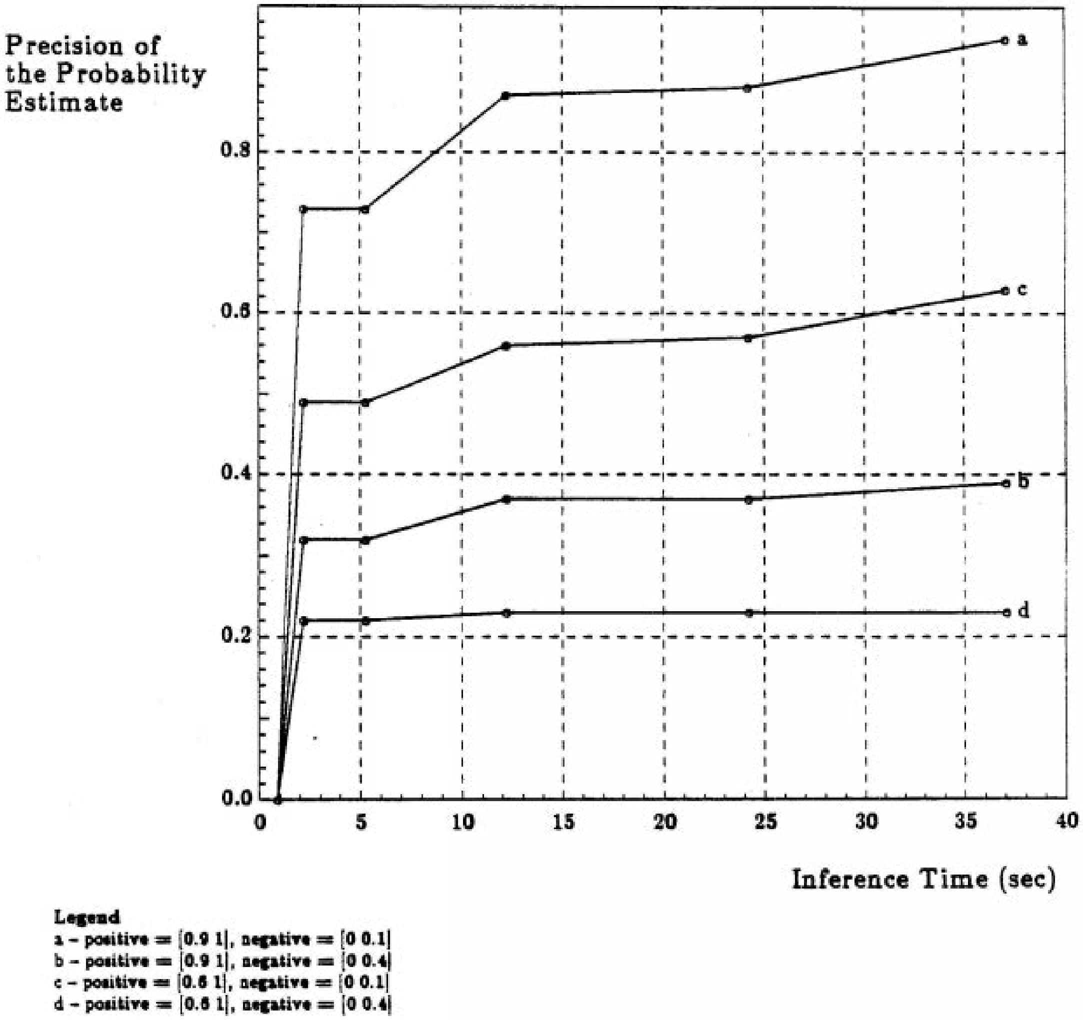
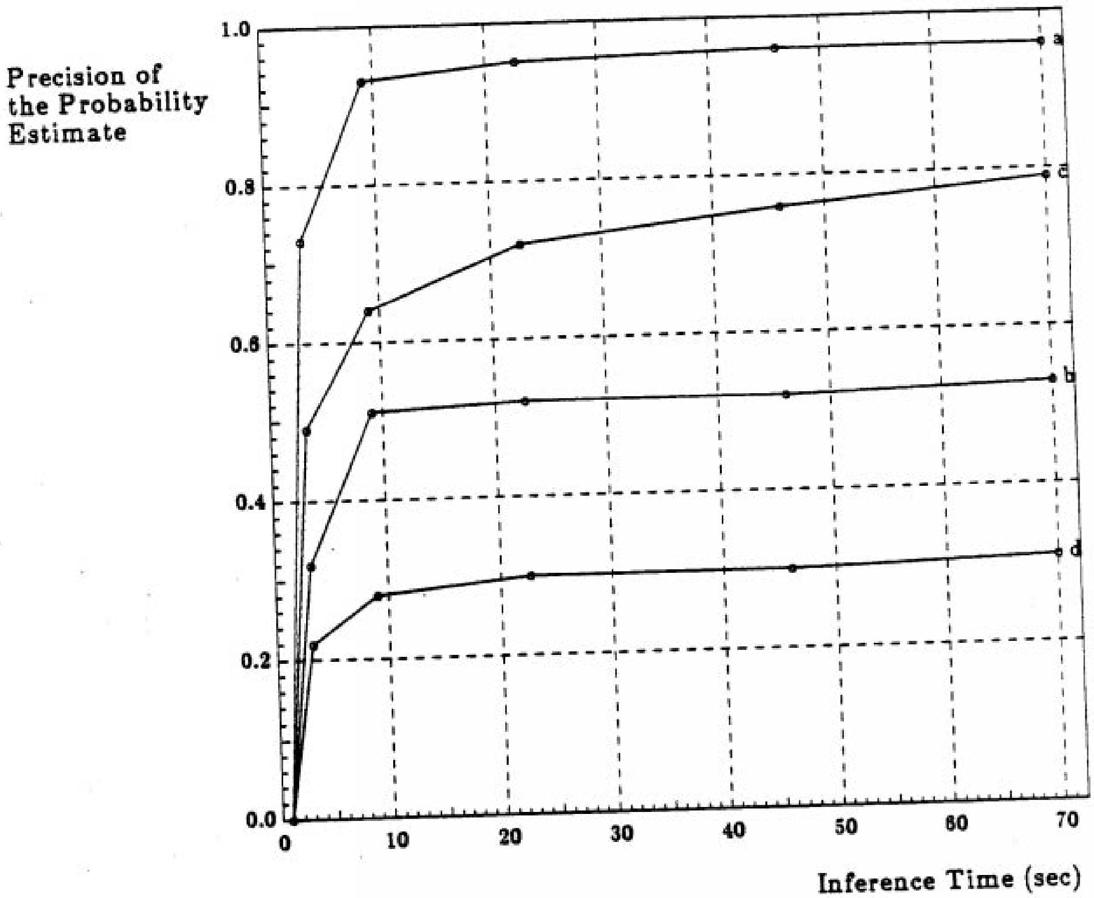


Figure 29. PPE(D) vs time for Rule Base 4.



Legend
a - positive = [0.9 1], negative = [0 0.1]
b - positive = [0.9 1], negative = [0 0.4]
c - positive = [0.6 1], negative = [0 0.1]
d - positive = [0.6 1], negative = [0 0.4]

Figure 30. PPE(D) vs time for Rule Base 5.

5.2. The ESTIMATOR System*

The problem of estimation is both time limited and value approximate. The goal of an estimation task is to produce the best possible answer to a question within a time frame which does not permit complete deliberation of every detail. The quality of an estimate is usually characterized by words like "rough" or "detailed" which refer to a range of precision of the estimate. The more precise an estimate is required to be, the more time is required to perform the inference. Thus the estimation task involves a direct tradeoff between precision and inference time.

Estimation tasks utilize two main classes of knowledge: knowledge pertaining to the task to be performed and knowledge which indicates the priority of the considerations in the task. The VPL formalism is ideally suited to representing this information because censored production rules allow both kinds of information to be embodied in a single representational structure. The rules represent approximate task level decisions and focus attention on the premises before the censors are considered.

Cost estimation is an area which illustrates many of the issues involved in estimation tasks in general. The ESTIMATOR system was developed with the goal of aiding an engineer in estimating the costs of installing heating ventilating and air conditioning (HVAC) systems in both new and existing buildings. The problem can be characterized as follows. The engineer is given a job specification which states the building type, size, required control systems, etc. and the goal is to determine the total cost of installing the required systems.

The problem can be divided into two main steps: equipment selection and cost calculation. The goal of the equipment selection step is to determine a list of equipment required for

* This work was done in collaboration with Jim Kelly of the Intelligent Systems Group.

For example, the following rule specifies the characteristics of a particular model of thermostat.

(Equipment = room-thermostat) & (Control = temperature) &
(Power = electric) & (Temperature-range = 20–120)
→ (Equipment-code = s1)
[(Accuracy = ±0.1) & (Durability = low) & (Delivery = 2-weeks)
: 1, 1, 0.7, 0.2

Equipment selection is done by first determining the types of equipment required such as thermostats, valves, controllers, etc. The system determines the probability that each type of equipment is required for the job. Since each equipment type represents an individual decision, the absolute probabilities are used. If the probability for a particular equipment type falls above a certain threshold, that equipment is determined to be required.

Next, for each type of equipment needed, the system then determines the model with the characteristics best suited to the job specification. For a particular equipment type, there will typically be numerous models with different combinations of characteristics. Since the equipment models are mutually exclusive, the decision is based on the relative probabilities of the suitability of each model to the job at hand. The probability of each model for an equipment type is determined and the model with the highest support is selected.

The substitution rules come into play in selection of the equipment models. If no equipment specification matches the requirements exactly, a piece of equipment may be chosen which exceeds the requirements. For example, if a thermostat of low accuracy is required, one of higher accuracy may be substituted. However, we only want to do this substitution if the desired equipment is unavailable because, in general, the higher accuracy device is more expensive, as mentioned earlier.

5.2.2. Calculation of the Bid

In the calculation step, the system uses five types of information:

- the job specification,
- the purchase price of each piece of equipment,
- the installation time for each piece of equipment,
- the number of zones, and
- markup adjustment rules.

The number of zones in the building is used as a multiplier to arrive at total equipment quantities. These quantities are multiplied by standard purchase and installation costs to arrive at a total cost estimate.

If our computer already in building *then* job is an extra.
If installing computer *then* extra potential is high.
If big job *then* extra potential is high.
If job location < 20 miles from office *then* markup–factor = 0.8
If job location > 40 miles from office *then* markup–factor = 1.2
If extra potential is high *then* markup–factor = 0.8
If extra to a previous job *then* markup–factor = 1.3
If no other competitive bids *then* markup–factor = 1.3
If labor intensive job *then* markup–factor = 1.2
If many outside purchases *then* markup–factor = 1.1
If good contractor *then* markup–factor = 0.8

Figure 31. Markup adjustment rules.

the particular job at hand. This is where most of the work is performed. The engineer must utilize large amounts of different types of information and be able to focus his attention on those aspects of the problem which most affect the cost. The goal of the second step is to determine the total equipment and labor cost for the job. In many cases this amounts to a relatively straight-forward calculation based on the number of zones in the building, the unit cost of each piece of equipment, and the labor required to install each piece of equipment.

5.2.1. Equipment Selection

The ESTIMATOR system uses four types of information to perform equipment selection:

- the job specification,
- engineering expertise,
- substitution rules, and
- equipment characteristics.

The job specification is the set of available facts describing the job. The specification may be incomplete in the sense that it does not provide information concerning all factors determining job cost. The engineering expertise represents the knowledge known by the engineer such as rules linking system type to required equipment type or building type to time schedule.

For example, the following rule states that schools are on "fast-track" building schedules unless the time of year is summer:

$$(\text{Building} = \text{school}) \rightarrow (\text{Schedule} = \text{fast-track})$$
$$\lfloor (\text{Time-of-year} = \text{summer}) : 1, 1, 0.75, 0.25$$

The substitution rules are a special type of engineering expertise; they indicate valid substitutions of one piece of equipment for another when something is not available. These rules allow approximate pattern matching to be done between the requirements and the equipment characteristics. The following set of rules specifies that a sensor of low accuracy may be replaced with one of higher accuracy (in $\pm$ degrees Fahrenheit). The greater the jump in accuracy, the lower the belief in the rule. This reflects the notion that a higher accuracy piece of equipment can be substituted for an acceptable lower accuracy piece but this is, in general, not a good practice.

$$(\text{Accuracy } \pm 1) \rightarrow (\text{Accuracy } \pm 0.5) : 0.6, 0.4$$
$$(\text{Accuracy } \pm 0.5) \rightarrow (\text{Accuracy } \pm 0.2) : 0.7, 0.3$$
$$(\text{Accuracy } \pm 0.2) \rightarrow (\text{Accuracy } \pm 0.1) : 0.8, 0.2$$

Notice that these rules form a chain from (Accuracy 1) to (Accuracy 0.1). The longer the chain needs to be in order to bridge the gap between the job requirement and the equipment characteristics, the lower the probability that the particular piece of equipment will satisfy our needs.

The equipment characteristics contain the information the engineer would obtain from equipment catalogs, such as the range of a sensor, the type of power it uses, the accuracy, etc.

A certain percentage profit markup is added to the total cost to produce the final bid. In determining this markup many additional heuristics may come into play: is there a potential for many "extras"?; is there a "good" general contractor on the job?; etc. For example, if the extra potential of a job is high then a company can afford to bid low on the contract, assuming that it will gain business in the future. As can be seen the answer to these and similar questions cannot be completely certain but nonetheless play an important role in the ultimate profitability. Using rules which represent these heuristics, an adjustment factor for the basic markup is derived and the bid is calculated. The markup adjustment rules are shown in figure 31. Adjusting the final bid in this way corresponds quite closely to the intuition and instincts relied upon by good estimators and salesmen.

5.2.3. Example Cost Estimation

This example shows the estimation of a heating system for a large office building. The job specification is shown in figure 32. The specification states that the system is to be powered pneumatically; the system should be of medium durability; the accuracy should be ± 0.2 degrees Fahrenheit; etc. Three estimates of varying quality are made: rough, average, and detailed. These differ in the amount of resources devoted to checking the provided conditions on the rules representing equipment characteristics. Figure 33 shows the calculation of a rough estimate for the bid on this job. The system determines the types of equipment required and selects the best equipment model for each, within the time available. The system configuration corresponding to the equipment types determined to be necessary is shown in figure 34. The possible range indicates that this selection is based purely on default information. The equipment type and model selection is done using the rule base in appendix 7.3.

(building office) : 1 1
(building-size large) : 1 1
(system heat) : 1 1
(power pneu) : 1 1
(durability med) : 1 1
(accuracy .2) : 1 1
(air-temp-range 50–80) : 1 1
(water-temp-range -50–150) : 1 1
(pressure-range 3–7) : 1 1
(control temp) : 1 1
(job construction) : 1 1
(equip-source existing) : 0 0
(system-purpose monitor) : 0 0

Figure 32. Job specification.

The total cost of the selected equipment is then calculated using the total purchase price and installation time. The markup is applied to this value to yield the bid. There are two factors affecting the markup: the job is large, and there are no competitive bids. The fact that that job is large decreases the markup and the lack of competition increases the markup. The result is a markup of 52% (2% over the standard). The final estimated bid is $45828.00, derived in 9.7 seconds.

The calculation of an average accuracy estimate for this job is shown in figure 35. The probability ranges on some of the equipment are now narrower and the system has now chosen both duct sensor S9 and pipe sensor S17 over S11 and S18 in the rough estimate respectively. The final bid is now $44612.00 and requires 14.6 seconds to calculate.

All details are taken into consideration in the detailed estimate shown in figure 36. The resulting bid is $43852.00. This differs from the rough estimate by only 4.5%, which shows

```
What quality estimate do you require (rough, average, detailed)? rough

What file contains the job specification? "ds-vpl>est-facts-2.lisp"
What are the markup adjustment factors?
((big-job) (no-competition))
How many zones are there? 20

Required equipment types:
    ROOM-THERMO
    DUCT-SENSOR
    PIPE-SENSOR
    VALVE
    PANEL-CONTROLLER

Chosen equipment models:
  S3   Most Likely Range = [0.50 0.60]  Possible Range = [0.00 1.00]
  S10  Most Likely Range = [0.64 0.92]  Possible Range = [0.00 1.00]
  S18  Most Likely Range = [0.70 0.80]  Possible Range = [0.00 1.00]
  V7   Most Likely Range = [0.50 0.60]  Possible Range = [0.00 1.00]
  RC6  Most Likely Range = [0.50 0.60]  Possible Range = [0.00 1.00]

Total purchase price is $25900.00
Total installation time is 170.0 hours
Total cost is $30150.00
Markup is 52.%
Bid is $45828.00

Estimation completed in 9.7 seconds elapsed time
```

Figure 33. Rough estimate.

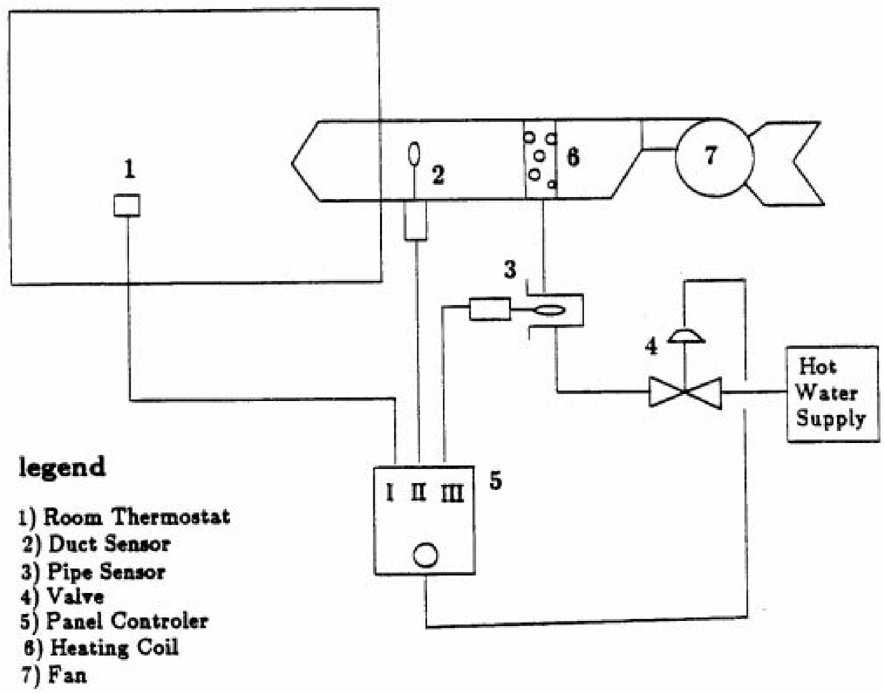


Figure 34. Schematic diagram of heating system.

Chosen equipment models:
S3 Most Likely Range = [0.50 0.60] Possible Range = [0.00 1.00]
S9 Most Likely Range = [0.56 0.84] Possible Range = [0.00 1.00]
S17 Most Likely Range = [0.72 0.75] Possible Range = [0.63 0.93]
V7 Most Likely Range = [0.40 0.48] Possible Range = [0.00 0.80]
RC6 Most Likely Range = [0.47 0.56] Possible Range = [0.00 0.93]

Total purchase price is $25100.00
Total installation time is 170.0 hours
Total cost is $29350.00
Markup is 52.%
Bid is $44612.00

Estimation completed in 14.6 seconds elapsed time

Figure 35. Average estimate.

Chosen equipment models:
S4 Most Likely Range = [0.74 0.78] Possible Range = [0.63 1.00]
S9 Most Likely Range = [0.56 0.84] Possible Range = [0.00 1.00]
S17 Most Likely Range = [0.72 0.75] Possible Range = [0.63 0.93]
V8 Most Likely Range = [0.74 0.78] Possible Range = [0.63 1.00]
RC6 Most Likely Range = [0.47 0.56] Possible Range = [0.00 0.93]

Total purchase price is $24600.00
Total installation time is 170.0 hours
Total cost is $28850.00
Markup is 52.%
Bid is $43852.00

Estimation completed in 17.5 seconds elapsed time

Figure 36. Detailed estimate.

that the rough estimate was quite close, considering that the prices of two similar pieces of equipment can differ by as much as 50%. The detailed estimate required 17.5 seconds. Comparing the rough estimate time of 9.7 seconds with this shows a 45% savings in inference time over the detailed estimate. These inference times are very low due to the small rule base being used. In an actual estimation system, the times would be more on the order of hours and the inference time saved in making rough versus detailed estimates would be significant. Note that the installation time remains constant for the three estimates. This is due to the fact that installation times are only dependent on the particular equipment type and the mode of powering it, i.e., pneumatic versus electric. The different qualities of estimates differ only in the effort they devote to checking the provided conditions on the rules but these two primary factors are represented as rule premises. Thus the installation times of the equipment models considered are all the same.

6. CONCLUSIONS

A system which can reason with incomplete and uncertain information under time constraints has been presented. In addition, the system is capable of reasoning efficiently with exception augmented rules. Controlled experiments using several knowledge bases were run to demonstrate the effects of knowledge base configuration on inference behavior. An application to the area of construction project cost estimation was shown.

6.1. System Performance

The VPL system represents a new kind of reasoning system capable of exploiting the tradeoff between inference time and inference precision. This is done by combining the ability to focus attention on determining the belief in those facts which are likely to provide the most information with the ability to reason without knowing the belief in those facts for which time did not permit consideration. The control information which allows the system to focus its attention is implicitly expressed in the natural form of censored production rules. The δ and γ values of the rules quantify this control information.

The resultant inference behavior is similar to that of default reasoning systems, where incomplete information is filled in by assumptions. In contrast to default reasoning, however, beliefs in the VPL system can have varying default character, based on the precision of the evidence considered. The extent to which a decision is based on default information is expressed in the difference between the possible and most likely ranges. The more dissimilar they are, the higher the default character of the inference.

The use of Dempster–Shafer theory to formalize VPL type inference provides a simple, intuitive notion of the precision of an inference which relates it to the amount of information found. This formalism allows the ignorance in the evidence to be preserved through the reasoning process and expressed in the decision. It was proved that the precision of inferences increases monotonically as a function of inference time. This is reflected in the width of the resulting Shafer interval. The VPL system can even reason with rules which have inconsistent censors since the system only uses as much information as is consistent with what is known.

The results of experiments on various knowledge base configurations show that significant savings in inference time can be realized at little expense in terms of precision. This is due to the fact that many knowledge bases produce inference behavior which conforms to a law of diminishing returns in terms of the precision of the inference. It was found that rule base structure has a greater effect on inference behavior than the precision of the evidence.

The success of the ESTIMATOR system shows that VPL can be highly effective as a tool for automating estimation tasks. The way in which the system focuses its attention on primary considerations mirrors very much the way in which human estimators work. The ability to use the uncertain inference to implement approximate pattern matching rules proved highly useful for the task of equipment selection.

The VPL system has significant implications for knowledge engineering in time critical domains in general. For applications in which censored production rules are a suitable representation the system frees a knowledge engineer from undue concern for inference times. The inference engine itself actually tailors the inference to fit the time constraints of the situation at hand. This also means that the system can adapt to situations which were unforeseen at the time of knowledge base design.

The VPL system complements machine learning systems. With earlier types of reasoning systems, there was no efficient way to reason with rules having many exceptions. When faced with an exception to a rule, learning systems had the choice of either ignoring it and accepting the fact that the rule would be applied incorrectly in exception cases or adding an exception to the rule and thereby decreasing the efficiency of reasoning with that rule in every future situation. The VPL system, on the other hand, allows the choice between using the approximate rule and using the exact rule to be delayed until inference time, when information regarding time constraints is available. In addition, the implications of this decision are clearly reflected in the default character of the decision.

6.2. Limitations and Future Research

The controlled experiments have shown that the inference behavior of the VPL system is quite sensitive to rule base structure. Prior knowledge of the rule base structure would allow more intelligent inference. For example, one rule base structure showed stepwise increases in the PPE of the decision with increasing inference time. An analysis of such a rule base could yield information indicating what inference time increments produce significant increases in the PPE.

The current VPL system performs best on knowledge bases which have information available at all levels of chaining. This structure produces a continuous increase in inference precision as more time is allotted. This limitation is due to the fact that the system uses a fixed breadth-first control strategy for censor chaining. A better approach would be to have the system learn the control strategy best suited to a particular knowledge base. The inference precision/time tradeoff inherent in VPL type inference serves as a handy utility measure on which search heuristics can be based. In order to optimize system performance, search should be ordered in such a way that the system maximizes the amount of information it gathers in a given amount of time. A learning component could determine a control strategy based on knowledge gained from past searches and knowledge of the combination and propagation functions. For example, this would lead to depth-first strategies for knowledge bases in which all the information was at the same depth; breadth-first strategies when information is evenly distributed along all search depths; and a combination of the two in other cases. The learning system would not have to be restricted to use with the approximate inference scheme employed in the current research but could be made general purpose by modularizing the knowledge of the combination and propagation functions of the particular uncertainty calculus being used.

The inference process of the VPL system does not currently take into consideration the certainty of the available evidence. Thus highly certain information which could have a large impact on a particular decision could be ignored if the time limit did not allow the backward-chaining search strategy to find it. This problem could be solved by allowing forward chaining inference to occur whenever a fact with high certainty enters the system. All implications of such facts would then be immediately available. The same argument holds for highly precise information.

Much world knowledge is best expressed in the form of taxonomies. Taxonomies carry more information than simple collections of rules. The concepts at any one level of the taxonomy are known to form a mutually exclusive spanning set. This information can be used, for example, to infer the truth of one concept at some level if all the other concepts at that level are known to be false. To make the system more effective, specialized combination and propagation functions for reasoning in taxonomies could be developed.

The current VPL system deals only with the tradeoff between computational cost and precision of inference. To exploit the tradeoff between computational cost and specificity, rules could be organized into abstraction hierarchies. Special control mechanisms could then choose the appropriate abstraction level based on the certainty required and time allotted. Inference at higher abstraction levels would be characterized by a higher certainty and a lower amount of required time.

The probabilistic logic formalism of Nilsson [1986] holds promise for the study of the tradeoff between between certainty and specificity. He formalizes the notion of the probability of a first order logic formula in terms of possible worlds. Using this formalism, the certainty of general statement can be determined given the certainties of more specific statements.

7. APPENDICES

7.1. User Manual

The VPL system is written in Common Lisp and runs on a Symbolics 3640. It consists of 1800 lines of code contained in the following seven files.

declarations.lisp – global declarations

data-base.lisp – knowledge base maintenance functions

infer.lisp – inference engine

encode.lisp – parser

top-level.lisp – user interface

unify.lisp – unification algorithm

analyze.lisp – rule base analyzer

The system is defined as a Common Lisp system and can be loaded with the function call (make-system 'vpl). Once loaded, the system is invoked by typing (vpl). Communication then proceeds via the user interface.

7.1.1. The User Interface

The user interface is a top level loop similar to that of prolog systems. Declarations, assertions, and commands may be entered in an interactive fashion. A summary of commands is shown below.

Command Summary

{} indicates a choice, [] indicates optional parameters
Keywords are written in all caps.

(lisp form) – evaluates the lisp form
(type name (elements)) – declare type
(pred predicate (argtypes) [:d (s p) :q (template) :proc {t/name}])
 – declare predicate
  d – default probability
  q – query user template, where (n) denotes the nth argument
  proc – name of attached procedure. t -> name is same as predicate
(assert {fact,rule}) – add to knowledge base
(retract RULE predicate) – retract all rules with a decision with the predicate
(retract FACT predicate) – retract all facts with this predicate
(show {RULES,FACTS,TIMES,FLAGS}) – show contents of knowledge base
 or the flags
(clear {RULES,FACTS,TIMES}) – clear the knowledge base
(load "filename") – take input from the file
(save "filename") – saves rules and facts in the file
(save-times "filename") – saves the TimeDB in the file
(load-times "filename") – loads the TimeDB from the file
(? (term time/depth)) – make a query in the specified time or
 to the specified depth. Time is a real while depth is an integer.
(? term) – make query with unlimited search
(analyze min/full) – analyze the rule base
 (user should clear all facts first)
end stop quit exit done bye – exit system

Commands may be entered in both list form as shown in the summary and in line form. Using the latter form, the command is typed separately on one line and the argument on the next line. The system stays in the submode for the particular command as long as arguments are being entered. This is a convenient way of entering numerous declarations or assertions in a row.

7.1.2. Representation

The VPL system uses a strongly typed language. Types and predicates must be declared before they are referred to. Types represent nominal domains for predicate arguments. The type declaration enumerates the elements of the domain. Types declarations may refer to other types. For example the following declarations are legal

(reptile (snake lizard))

(bird (robin sparrow))

(animal ((reptile) (bird) dog cat horse)

Predicates are declared by listing the predicate name, followed by the argument types, followed by optional parameters. The parameters are

:d (s p) – default probability to be used if value is [0 1]

:q (template) – user query template

:proc name – name of attached procedure

The query template allows the user to specify an English translation of the predicate. Rather than looking in the data base for the value of this predicate, the system will query the user using the template. A template entry of (n) is substituted with value of the nth argument at run time.

The procedure name is a lisp function which is called to return the value of this predicate. If name = t, the procedure name is the same as the predicate name. The following declaration specifies a default and a query template for the predicate "mother."

(Mother (person female) :d (.8 .9) :q (Is (2) the mother of (1)))

Once the types and predicates are defined, rules and facts using these may be asserted. Rules consist of a conjunction of premises, a single decision, and a disjunction of censors. Unless rules are written as

((p1 $x)(p2 $y)... => (d $x $y) ⌊ (c1 a)(c2 $y)... $\gamma^+ \gamma^- \delta^+ \delta^-$)

and provided rules as

((p1 $x)(p2 $y)... => (d $x $y) ⌈ (c1 a)(c2 $y)... $\gamma^+ \gamma^- \delta^+ \delta^-$)

where $x and $y denote variables. The unless operator is defined in the vpl font as the key symbol–h and the provided operator as symbol–g.

Facts are written as ((P $x a) s p).

7.2. Experimental Data

7.2.1. Rule Base 1

premises-per-rule 1

censors-per-rule 1

premises-to-chain-on 1

censors-to-chain-on 1

rules-per-premise 2

rules-per-censor 2

premise-rules-to-chain-on 1

censor-rules-to-chain-on 1

depth 5

((P0002) => (D) ⌊ (C0003) 1.0 1.0 0.8 0)
((P0004) => (D) ⌊ (C0005) 1.0 1.0 0.8 0)
((P0006) => (P0002) ⌊ (C0007) 1.0 1.0 0.8 0)
((P0008) => (P0002) ⌊ (C0009) 1.0 1.0 0.8 0)
((P0010) => (C0003) ⌊ (C0011) 1.0 1.0 0.8 0)
((P0012) => (C0003) ⌊ (C0013) 1.0 1.0 0.8 0)
((P0014) => (P0006) ⌊ (C0015) 1.0 1.0 0.8 0)
((P0016) => (P0006) ⌊ (C0017) 1.0 1.0 0.8 0)
((P0018) => (P0010) ⌊ (C0019) 1.0 1.0 0.8 0)
((P0020) => (P0010) ⌊ (C0021) 1.0 1.0 0.8 0)
((P0022) => (C0011) ⌊ (C0023) 1.0 1.0 0.8 0)
((P0024) => (C0011) ⌊ (C0025) 1.0 1.0 0.8 0)
((P0026) => (C0007) ⌊ (C0027) 1.0 1.0 0.8 0)
((P0028) => (C0007) ⌊ (C0029) 1.0 1.0 0.8 0)

((P0030) => (P0014) ⌊ (C0031) 1.0 1.0 0.8 0)

((P0032) => (P0014) ⌊ (C0033) 1.0 1.0 0.8 0)

((P0034) => (P0018) ⌊ (C0035) 1.0 1.0 0.8 0)

((P0036) => (P0018) ⌊ (C0037) 1.0 1.0 0.8 0)

((P0038) => (P0022) ⌊ (C0039) 1.0 1.0 0.8 0)

((P0040) => (P0022) ⌊ (C0041) 1.0 1.0 0.8 0)

((P0042) => (P0026) ⌊ (C0043) 1.0 1.0 0.8 0)

((P0044) => (P0026) ⌊ (C0045) 1.0 1.0 0.8 0)

((P0046) => (C0027) ⌊ (C0047) 1.0 1.0 0.8 0)

((P0048) => (C0027) ⌊ (C0049) 1.0 1.0 0.8 0)

((P0050) => (C0023) ⌊ (C0051) 1.0 1.0 0.8 0)

((P0052) => (C0023) ⌊ (C0053) 1.0 1.0 0.8 0)

((P0054) => (C0019) ⌊ (C0055) 1.0 1.0 0.8 0)

((P0056) => (C0019) ⌊ (C0057) 1.0 1.0 0.8 0)

((P0058) => (C0015) ⌊ (C0059) 1.0 1.0 0.8 0)

((P0060) => (C0015) ⌊ (C0061) 1.0 1.0 0.8 0)

((P0062) => (P0030) ⌊ (C0063) 1.0 1.0 0.8 0)

((P0064) => (P0030) ⌊ (C0065) 1.0 1.0 0.8 0)

((P0066) => (P0034) ⌊ (C0067) 1.0 1.0 0.8 0)

((P0068) => (P0034) ⌊ (C0069) 1.0 1.0 0.8 0)

((P0070) => (P0038) ⌊ (C0071) 1.0 1.0 0.8 0)

((P0072) => (P0038) ⌊ (C0073) 1.0 1.0 0.8 0)

((P0074) => (P0042) ⌊ (C0075) 1.0 1.0 0.8 0)

((P0076) => (P0042) ⌊ (C0077) 1.0 1.0 0.8 0)

((P0078) => (P0046) ⌊ (C0079) 1.0 1.0 0.8 0)

((P0080) => (P0046) ⌊ (C0081) 1.0 1.0 0.8 0)

((P0082) => (P0050) ⌊ (C0083) 1.0 1.0 0.8 0)

((P0084) => (P0050) ⌊ (C0085) 1.0 1.0 0.8 0)

((P0086) => (P0054) ⌊ (C0087) 1.0 1.0 0.8 0)

((P0088) => (P0054) ⌊ (C0089) 1.0 1.0 0.8 0)

((P0090) => (P0058) ⌊ (C0091) 1.0 1.0 0.8 0)

((P0092) => (P0058) ⌊ (C0093) 1.0 1.0 0.8 0)

((P0094) => (C0059) ⌊ (C0095) 1.0 1.0 0.8 0)

((P0096) => (C0059) ⌊ (C0097) 1.0 1.0 0.8 0)

((P0098) => (C0055) ⌊ (C0099) 1.0 1.0 0.8 0)

((P0100) => (C0055) ⌊ (C0101) 1.0 1.0 0.8 0)

((P0102) => (C0051) ⌊ (C0103) 1.0 1.0 0.8 0)

((P0104) => (C0051) ⌊ (C0105) 1.0 1.0 0.8 0)

((P0106) => (C0047) ⌊ (C0107) 1.0 1.0 0.8 0)

((P0108) => (C0047) ⌊ (C0109) 1.0 1.0 0.8 0)

((P0110) => (C0043) ⌊ (C0111) 1.0 1.0 0.8 0)

((P0112) => (C0043) ⌊ (C0113) 1.0 1.0 0.8 0)

((P0114) => (C0039) ⌊ (C0115) 1.0 1.0 0.8 0)

((P0116) => (C0039) ⌊ (C0117) 1.0 1.0 0.8 0)

((P0118) => (C0035) ⌊ (C0119) 1.0 1.0 0.8 0)

((P0120) => (C0035) ⌊ (C0121) 1.0 1.0 0.8 0)

((P0122) => (C0031) ⌊ (C0123) 1.0 1.0 0.8 0)

((P0124) => (C0031) ⌊ (C0125) 1.0 1.0 0.8 0)

((P0004) 0.6 1) ((P0008) 0.6 1) ((P0012) 0.6 1) ((P0016) 0.6 1) ((P0020) 0.6 1) ((P0024) 0.6 1) ((P0028) 0.6 1) ((P0032) 0.6 1) ((P0036) 0.6 1) ((P0040) 0.6 1) ((P0044) 0.6 1) ((P0048) 0.6 1) ((P0052) 0.6 1) ((P0056) 0.6 1) ((P0060) 0.6 1) ((P0064) 0.6 1) ((P0068) 0.6 1) ((P0072) 0.6 1) ((P0076) 0.6 1) ((P0080) 0.6 1) ((P0084) 0.6 1) ((P0088) 0.6 1) ((P0092) 0.6 1) ((P0096) 0.6 1) ((P0100) 0.6 1) ((P0104) 0.6 1) ((P0108) 0.6 1) ((P0112) 0.6 1) ((P0116) 0.6 1) ((P0120) 0.6 1) ((P0124) 0.6 1) ((P0062) 0.6 1) ((P0066) 0.6 1) ((P0070) 0.6 1) ((P0074) 0.6 1) ((P0078) 0.6 1) ((P0082) 0.6 1) ((P0086) 0.6 1) ((P0090) 0.6 1) ((P0094) 0.6 1) ((P0098) 0.6 1) ((P0102) 0.6 1) ((P0106) 0.6 1) ((P0110) 0.6 1) ((P0114) 0.6 1) ((P0118) 0.6 1) ((P0122) 0.6 1) ((C0125) 0.6 1) ((C0121) 0.6 1) ((C0117) 0.6 1) ((C0113) 0.6 1) ((C0109) 0.6 1) ((C0105) 0 0.1) ((C0101) 0 0.1) ((C0097) 0 0.1) ((C0093) 0 0.1) ((C0089) 0 0.1) ((C0085) 0 0.1) ((C0081) 0 0.1) ((C0077) 0 0.1) ((C0073) 0 0.1) ((C0069) 0 0.1) ((C0065) 0 0.1) ((C0061) 0.6 1) ((C0057) 0.6 1) ((C0053) 0.6 1) ((C0049) 0 0.1) ((C0045) 0 0.1) ((C0041) 0 0.1) ((C0037) 0 0.1) ((C0033) 0 0.1) ((C0029) 0.6 1) ((C0025) 0 0.1) ((C0021) 0 0.1) ((C0017) 0 0.1) ((C0013) 0.6 1) ((C0009) 0 0.1) ((C0005) 0 0.1) ((C0123) 0.6 1) ((C0119) 0.6 1) ((C0115) 0.6 1) ((C0111) 0.6 1) ((C0107) 0.6 1) ((C0103) 0 0.1) ((C0099) 0 0.1) ((C0095) 0 0.1) ((C0091) 0 0.1) ((C0087) 0 0.1) ((C0083) 0 0.1) ((C0079) 0 0.1) ((C0075) 0 0.1) ((C0071) 0 0.1) ((C0067) 0 0.1) ((C0063) 0 0.1)

7.3. ESTIMATOR Knowledge Base

TYPE

(building-type (medical office military school))

(time-frame (slow average fast-track))

(system-type (heat a/c vav controlers))

(equip-type (room-thermo valve duct-sensor pipe-sensor panel-controler field-controler))

(thermostats (s1 s2 s3 s4 s5))

(valves (v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12))

(sensors (s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17 s18))

(controlers (rc1 rc2 rc3 rc4 rc5 rc6 rc7 rc8 rc9 rc10 rc11 rc12))

(equip-codes ((thermostats) (sensors) (valves) (controlers)))

(locations (room duct pipe panel field))

(controls (temp humidity pressure air-flow))

(mediums (air water))

(powers (elect pneu))

(ranges (1-5 3-7 8-13 50-80 20-120 0-100 -50-150 40-240))

(accuracies (.1 .2 .5 1)) ; REQUIRED ACCURACY IN +/- DEGREES FARENHEIGHT

(level (low med high))

(delivery-time (1 2 3 4)) ; REQUIRED DELIVERY TIME IN WEEKS

(costs (50 75 100 125))

(season (summer winter spring fall))

(source (existing incompany outside))

(job-type (construction service management))

(purpose-type (control monitor))

(cooling-type (water DX))

(heating-type (water electric))

(size (small medium large))

PRED

(building (building-type))

(schedule (time-frame))

(system (system-type))

(location (locations))

(control (controls))

(medium (mediums))

(power (powers))

(air-temp-range (ranges))

(water-temp-range (ranges))

(pressure-range (ranges))

(equip-code (equip-codes))

(accuracy (accuracies))

(durability (level))

(delivery (delivery-time))

(time-of-year (season))

(building-size (size))

(heating (heating-type))

(cooling (cooling-type))

(job (job-type))

(equip-source (source))

(system-purpose (purpose-type))

(equip (equip-type))

ASSERT

((building-size large) => (heating water) .8 .1)

((building-size large) => (cooling water) .7 .1)

((building-size small) => (heating electric) .6 .2)

((building-size small) => (cooling DX) .7 .2)

```
( (job service) => (equip-source existing) .7 .1)
( (job management) => (equip-source existing) .7 .2)
( (job construction) => (~ equip-source existing) .8 .1)

; DETERMINE EQUIPMENT TYPE BASED ON SYSTEM TYPE
( (system a/c) => (equip room-thermo)
  ⌊ (equip-source existing) (system-purpose monitor) 1 1 .8 .1)
( (system a/c) => (equip duct-sensor) ⌊ (equip-source existing) 1 1 .7 .2)
( (system a/c) => (equip pipe-sensor) ⌊ (equip-source existing) (cooling DX) 1 1 .6 .2)
( (system a/c) => (equip valve)
  ⌊ (equip-source existing) (system-purpose monitor) (cooling DX) 1 1 .5 .3)
( (system a/c) => (equip panel-controler)
  ⌊ (equip-source existing) (system-purpose monitor) 1 1 .7 .2)

( (system heat) => (equip room-thermo)
  ⌊ (equip-source existing) (system-purpose monitor) 1 1 .9 .1)
( (system heat) => (equip duct-sensor) ⌊ (equip-source existing) 1 1 .7 .1)
( (system heat) => (equip pipe-sensor) ⌊ (equip-source existing) (heating electric) 1 1 .7 .2)
( (system heat) => (equip valve)
  ⌊ (equip-source existing) (system-purpose monitor) (heating electric) 1 1 .6 .2)
( (system heat) => (equip panel-controler)
  ⌊ (equip-source existing) (system-purpose monitor) 1 1 .8 .1)

; TIME SCHEDULE RULES
( (building medical) => (schedule fast-track) .9 0)
( (building office) => (schedule slow) .7 .3)
( (building military) => (schedule average) .7 .3)
( (building school) => (schedule fast-track) ⌊ (time-of-year summer) 1 1 .75 .25)

( (schedule slow) => (delivery 4) .9 .1)
( (schedule average) => (delivery 2) .9 .1)
```

```
( (schedule fast-track) => (delivery 1) .9 .1)

; MEDIUM & LOCATION RULES

( (system heat) => (medium air) .9 0)
( (system heat) => (medium water) .7 .1)
( (system a/c) => (medium air) .9 0)
( (system a/c) => (medium water) .8 .2)
( (system vav) => (medium air) .9 0)

( (medium air) => (location duct) .8 .1)
( (medium air) => (location room) .8 .1)
( (medium water) => (location pipe) .9 0)

; ---------- SUBSTITUTION RULES ----------
; These rules indicate how good a substitution of one piece of equipment for
; another is.

; IF A SENSOR IN THE RANGE 50-80 IS NEEDED THEN ONE IN THE RANGE 20-120 IS ALSO
; SOMEWHAT ACCEPTABLE.
( (air-temp-range 50-80) => (air-temp-range 20-120) .8 .2)
( (water-temp-range 0-100) => (water-temp-range -50-150) .8 .2)

; WE CAN REPLACE A SENSOR OF LOW ACCURACY WITH ONE OF HIGHER ACCURACY.
; IN GENERAL WE DON'T WANT TO DO THIS BECAUSE WE PAY FOR ACCURACY.
( (accuracy  1) => (accuracy .5) .8 .2)
( (accuracy .5) => (accuracy .2) .7 .3)
( (accuracy .2) => (accuracy .1) .8 .2)

; REPLACE A PIECE OF EQUIPMENT OF LOW DURABILITY WITH ONE OF HIGHER DURABILITY.
( (durability low) => (durability med) .8 .2)
( (durability med) => (durability high) .8 .2)

( (delivery 4) => (delivery 3) 1 0)
```

```
( (delivery 3) => (delivery 2) 1 0)
( (delivery 2) => (delivery 1) 1 0)

; ———— THERMOSTATS ————
( (equip room-thermo) (control temp) (power elect) (air-temp-range 20-120) =>
  (equip-code s1) [ (accuracy .1) (durability low) (delivery 2) 1 1 .7 .2)

( (equip room-thermo) (control temp) (power elect) (air-temp-range 20-120) =>
  (equip-code s2) [ (accuracy .5) (durability med) (delivery 1) 1 1 .3 .6)

( (equip room-thermo) (control temp) (power pneu) (air-temp-range 50-80) =>
  (equip-code s3) [ (accuracy 1) (durability med) (delivery 1) 1 1 .5 .4)

( (equip room-thermo) (control temp) (power pneu) (air-temp-range 50-80) =>
  (equip-code s4) [ (accuracy .2) (durability med) (delivery 3) 1 1 .3 .6)

( (equip room-thermo) (control temp) (power pneu) (air-temp-range 50-80) =>
  (equip-code s5) [ (accuracy .1) (durability high) (delivery 4) 1 1 .2 .7)

; ———— DUCT SENSORS ————
( (equip duct-sensor) (control temp) (power elect) (air-temp-range 20-120) =>
  (equip-code s6) [ (accuracy .1) (durability high) (delivery 3) 1 1 .8 .1)

( (equip duct-sensor) (control temp) (power elect) (air-temp-range 50-80) =>
  (equip-code s7) [ (accuracy .2) (durability med) (delivery 4) 1 1 .6 .3)

( (equip duct-sensor) (control temp) (power elect) (air-temp-range 50-80) =>
  (equip-code s8) [ (accuracy .2) (durability med) (delivery 2) 1 1 .4 .5)

( (equip duct-sensor) (control temp) (power pneu) (air-temp-range 20-120) =>
  (equip-code s9) [ (accuracy .5) (durability med) (delivery 2) 1 1 .7 .2)

( (equip duct-sensor) (control temp) (power pneu) (air-temp-range 20-120) =>
```

```
(equip-code s10) [ (accuracy .1) (durability high) (delivery 4) 1 1 .8 .1)

( (equip duct-sensor) (control temp) (power pneu) (air-temp-range 50-80) =>
 (equip-code s11) [ (accuracy .1) (durability high) (delivery 1) 1 1 .6 .3)

; ———— PIPES SENSORS ————
( (equip pipe-sensor) (control temp) (power elect) (water-temp-range 0-100) =>
 (equip-code s12) [ (accuracy .5) (durability low) (delivery 1) 1 1 .8 .1)

( (equip pipe-sensor) (control temp) (power elect) (water-temp-range -50-150) =>
 (equip-code s13) [ (accuracy .2) (durability low) (delivery 1) 1 1 .6 .3)

( (equip pipe-sensor) (control temp) (power elect) (water-temp-range -50-150) =>
 (equip-code s14) [ (accuracy .1) (durability med) (delivery 1) 1 1 .4 .5)

( (equip pipe-sensor) (control temp) (power elect) (water-temp-range 0-100) =>
 (equip-code s15) [ (accuracy .1) (durability high) (delivery 2) 1 1 .8 .1)

( (equip pipe-sensor) (control temp) (power pneu) (water-temp-range 0-100) =>
 (equip-code s16) [ (accuracy .2) (durability low) (delivery 1) 1 1 .8 .1)

( (equip pipe-sensor) (control temp) (power pneu) (water-temp-range -50-150) =>
 (equip-code s17) [ (accuracy .2) (durability med) (delivery 4) 1 1 .3 .6)

( (equip pipe-sensor) (control temp) (power pneu) (water-temp-range -50-150) =>
 (equip-code s18) [ (accuracy 1) (durability high) (delivery 2) 1 1 .7 .2)

; VALVES
( (equip valve) (control temp) (power elect) (pressure-range 1-5) =>
 (equip-code v1) [ (accuracy .1) (durability low) (delivery 2) 1 1 .2 .7)

( (equip valve) (control temp) (power elect) (pressure-range 1-5) =>
 (equip-code v2) [ (accuracy .1) (durability med) (delivery 3) 1 1 .3 .6)
```

```
( (equip valve) (control temp) (power elect) (pressure-range 1-5) =>
  (equip-code v3) [ (accuracy .1) (durability high) (delivery 4) 1 1 .1 .8)

( (equip valve) (control temp) (power elect) (pressure-range 1-5) =>
  (equip-code v4) [ (accuracy 1) (durability low) (delivery 2) 1 1 .1 .8)

( (equip valve) (control temp) (power elect) (pressure-range 1-5) =>
  (equip-code v5) [ (accuracy 1) (durability med) (delivery 3) 1 1 .1 .8)

( (equip valve) (control temp) (power elect) (pressure-range 1-5) =>
  (equip-code v6) [ (accuracy 1) (durability high) (delivery 3) 1 1 .2 .7)


( (equip valve) (control temp) (power pneu) (pressure-range 3-7) =>
  (equip-code v7) [ (accuracy .1) (durability low) (delivery 2) 1 1 .5 .4)

( (equip valve) (control temp) (power pneu) (pressure-range 3-7) =>
  (equip-code v8) [ (accuracy .2) (durability med) (delivery 3) 1 1 .3 .6)

( (equip valve) (control temp) (power pneu) (pressure-range 3-7) =>
  (equip-code v9) [ (accuracy 1) (durability high) (delivery 1) 1 1 .2 .7)


( (equip valve) (control temp) (power pneu) (pressure-range 8-13) =>
  (equip-code v10) [ (accuracy .1) (durability low) (delivery 4) 1 1 .2 .7)

( (equip valve) (control temp) (power pneu) (pressure-range 8-13) =>
  (equip-code v11) [ (accuracy .1) (durability med) (delivery 4) 1 1 .3 .6)

( (equip valve) (control temp) (power pneu) (pressure-range 8-13) =>
  (equip-code v12) [ (accuracy 1) (durability high) (delivery 2) 1 1 .5 .4)


; CONTROLERS
( (equip panel-controler) (control temp) (power elect) =>
  (equip-code rc1) [ (accuracy .1) (durability low) (delivery 2) 1 1 .2 .7)
```

((equip panel–controler) (control temp) (power elect) =>
(equip–code rc2) [(accuracy 1) (durability med) (delivery 3) 1 1 .3 .6)

((equip panel–controler) (control temp) (power elect) =>
(equip–code rc3) [(accuracy .1) (durability high) (delivery 4) 1 1 .5 .4)

((equip panel–controler) (control temp) (power pneu) =>
(equip–code rc4) [(accuracy .1) (durability low) (delivery 3) 1 1 .2 .7)

((equip panel–controler) (control temp) (power pneu) =>
(equip–code rc5) [(accuracy .5) (durability med) (delivery 4) 1 1 .3 .6)

((equip panel–controler) (control temp) (power pneu) =>
(equip–code rc6) [(accuracy 1) (durability med) (delivery 4) 1 1 .5 .4)

((equip field–controler) (control temp) (power elect) =>
(equip–code rc7) [(accuracy .1) (durability med) (delivery 2) 1 1 .2 .7)

((equip field–controler) (control temp) (power elect) =>
(equip–code rc8) [(accuracy 1) (durability high) (delivery 3) 1 1 .3 .6)

((equip field–controler) (control temp) (power elect) =>
(equip–code rc9) [(accuracy .2) (durability high) (delivery 4) 1 1 .5 .4)

((equip field–controler) (control temp) (power pneu) =>
(equip–code rc10) [(accuracy .1) (durability med) (delivery 3) 1 1 .2 .7)

((equip field–controler) (control temp) (power pneu) =>
(equip–code rc11) [(accuracy .5) (durability high) (delivery 4) 1 1 .3 .6)

((equip field–controler) (control temp) (power pneu) =>
(equip–code rc12) [(accuracy 1) (durability high) (delivery 2) 1 1 .5 .4)

REFERENCES

Barnett,J.A. Computational Methods for a Mathematical Theory of Evidence, Proceedings IJCAI-81, August, 1981.

Becker,J.M. Inductive Learning of Decision Rules with Exceptions: Methodology and Experimentation, Tech. Report ISG 85-14, Intelligent Systems Group, Dept. of Computer Science, Univ. of Illinois at Urbana-Champaign, 1985.

Dempster,A.P. A Generalization of Bayesian Inference, Journal of the Royal Statistical Society, vol 30, no 2, 1968.

Dubois,D.,Prade,H. Combination and Propogation of Uncertainty with Belief functions, Proceedings IJCAI-85, August, 1985.

Duda, R.O., Hart,P.E.,Konolige,K.,Reboh,R. A Computer-based Consultant for Mineral Exploration, Final Report, SRI Project 6415, Sept, 1979.

Duda, R.O., Hart, P.E., and Nilsson, N.J. Subjective Bayesian Methods For Rule-Based Inference Systems, S.R.I. T.N. 124, A.I. Center, S.R.I., January, 1976.

Garvey,T.D.,Lowrance,J.D.,Fischler,M.A. An Inference Technique for Integrating Knowledge from Disparate Sources, Proceedings IJCAI-81, August, 1981.

Ginsberg,M.L. Non-monotonic Reasoning Using Dempster's Rule, Proceedings AAAI-84, August, 1984.

Haddawy,P. Implementation of and Experiments with a Variable Precision Logic Inference System, Proceedings AAAI-86, August, 1986.

Michalski,R.S. Knowledge Repair Mechanisms: Evolution vs Revolution, Tech. Report ISG 85-15, Intelligent Systems Group, Dept. of Computer Science, Univ. of Illinois at Urbana-Champaign, 1985.

Michalski,R.S., Winston,P.H. Variable Precision Logic, Artificial Intelligence Vol. 29, No. 2, August, 1986.

Nilsson,N.J. Probabilistic Logic, Artificial Intelligence Vol. 28, 1986.

Reiter,R. A Logic for Default Reasoning, Artificial Intelligence Vol. 13, No. 1,2, 1980.

Shafer,G. A Mathematical Theory of Evidence, Princeton Univ Press, Princeton, 1976.

Shortliffe, E.H. Computer-Based Medical Consultations: MYCIN, School of Medicine, Stanford University, American Elsevier, 1976.

Shortliffe, E.H., Buchanan,B.G. A Model of Inexact Reasoning in Medicine. In: Rule-based Expert Systems, ed. Buchanan,B.G., Shortliffe,E.H., Addison-Wesley, Reading, MA, 1984.

Winston,P.H. Learning by Augmenting Rules and Accumulating Censors. In: Machine Learning II, ed. Michalski,R.S., Carbonell,J.G., Mitchell,T.M., Morgan Kaufmann, Los Altos, CA, 1986.

Wise,B.P.,Henrion,B. A framework for comparing uncertain inference systems to probability, Proceedings of the AAAI Workshop on Uncertainty and Probability in Artificial Intelligence, Menlo Park, CA, August, 1984.

BIBLIOGRAPHIC DATA SHEET	1. Report No. UIUCDCS-F-86-959	2.	3. Recipient's Accession No.
4. Title and Subtitle A Variable Precision Logic Inference System Employing the Dempster-Shafer Uncertainty Calculus			5. Report Date December 1986 6. ISG 86-11
7. Author(s)			8. Performing Organization Rept. No.
9. Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, IL 61801			10. Project/Task/Work Unit No. 11. Contract/Grant No. ONR N00014-82-K-0186 NSF DCR 89-06801 DARPA N00014-K-85-0878
12. Sponsoring Organization Name and Address			13. Type of Report & Period Covered 14.

15. Supplementary Notes

16. Abstracts

This thesis presents the theory and implementation of a method of reasoning with incomplete and uncertain information under time constraints called Variable Precision Logic. The Variable Precision Logic Inference System varies the precision of its inferences to produce a decision within a specified time limit. Knowledge is represented as Censored Production Rules and inference is performed using a scheme based on Dempster-Shafer theory. The method is a form of default reasoning which produces decisions with a varying amount of default character, based on the amount of available information. A series of experiments using various knowledge bases shows the effect of rule base configuration on system behavior. An application to the problem of construction project cost estimation is described.

17. Key Words and Document Analysis. 17a. Descriptors

Approximate inference
Dempster-Shafer theory
Real time inference
Censored production rules
Expert systems
Default reasoning

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 108
	20. Security Class (This Page) UNCLASSIFIED	22. Price

FORM NTIS-35 (10-70) USCOMM-DC 40329-P71