CONTENTS

TUTORIAL PAPERS
2-15 Plausible Reasoning
   J.L. O'NEILL

RESEARCH PAPERS
16-24 The Transition to Fifth Generation Technology:
   Conceptual Schema Implementation
   G.M. McGrath

25-31 Integrating the Structured Analysis and Design Models:
   a Category-Theoretic Approach
   T.M. Tse

32-41 The Use of Non-hierarchical Allocation Methods for
   Clustering Large Sets of Data
   L. Belbin

SPECIAL FEATURES
1 Editorial
15 Helpers Needed
24 Call for Papers for ACC87-MBA
31 Call for Papers for Special Issue on Digital Image Processing
41 Call for Papers for 18th Annual CAE Computer Conference
42-46 Book Reviews
47-48 Letter to the Editor and Reply

Published for Australian Computer Society Incorporated
Registered by Australia Post, Publication No. NBG 1124
NonStop EXT.
The Computer Room on Wheels.

Everything; Processors, Memory, Disk Drives, Tape Cartridge, Communications and I/O Controllers, is contained in one cabinet that needs no special air-conditioned environment. Wheel it in. Plug it in. Unpacking and set up takes only about 1 hour.

YOU GROW,
EXT GROWS
To add power, just add processors. You can grow from 4 to over 22 ET-1 transactions per second. Memory can be upgraded from 4 to 64 megabytes. Grow at each location or grow to new locations, with software that's compatible throughout your Tandem network.

LOCAL AUTONOMY,
CORPORATE UNITY
The EXT is the low cost system that lets you distribute your processing to the local level. More work gets done and you save communications costs. And because the EXT knows its way through the major protocols (SNA, X.25, MAP and OSI) it can connect to and integrate with your present network.

THE DATA EXPRESSWAY
Multiple high density disk drives allow parallel data access. This speeds up data flow and shortens response time. Each drive can be individually maintained and replaced without interrupting service.

NO MORE TAPE REELS
A Tape Cartridge snaps in and out, storing 90 megabytes, replacing two full reels of tape. There's room to add another tape drive if you need it.

EVERYTHING'S EASIER
All major components are field replaceable, so servicing them is fast and foolproof. Our high-performance, relational database simplifies use. And powerful programmer tools speed up application development.

NO FAULT INSURANCE
Tandem's unique, parallel architecture keeps you up and running without the waste of idle backup components. Instead, multiple components share the workload. If one fails, the others pick up the slack.

TANDEM COMPUTERS
MELBOURNE
(03) 268 2900
ADELAIDE
(08) 211 8665
SYDNEY
(02) 957 8586
PERTH
(09) 325 1588
CANBERRA
(062) 49 1471
WELLINGTON (N.Z.)
(4) 72 3286
BRISBANE
(07) 369 4511
MELBOURNE
(03) 268 2900
ADELAIDE
(08) 211 8665
SYDNEY
(02) 957 8586
PERTH
(09) 325 1588
CANBERRA
(062) 49 1471
WELLINGTON (N.Z.)
(4) 72 3286
Editorial

As you were! In spite of the announcement in the November 1986 issue, the journal does not have a new editor, at least not yet. The editor-designate, Dr Barbara Kidman of the University of Adelaide found in December that, for personal reasons, she would be unable to take the position. Her decision has been received with regret, and she has written a letter that appears on page 47 along with other correspondence. The previous editor is continuing for a little longer, and unexpectedly writing yet another editorial. It is hoped that a further announcement on the editorial succession will be possible shortly.

In the paper by L. Belbin Figure 10 (page 40) has been reproduced in colour on the inside back cover. This arrangement of material within the issue has been occasioned by one of the Journal’s infrequent forays into colour printing and a desire to minimise the consequent additional printing costs. Improvements in printing technology, and the use of unpaid editorial labour have allowed the journal’s production costs to remain remarkably steady over the last five years. However postage costs have been rising inexorably, and the volume of editorial activity has been increasing steadily, necessitating more paid professional help. With regret we must announce a rise in the Journal’s subscription price, from $20 to $25. (The last price rise, from $18 to $20, occurred at the end of 1981. May the next one wait just as long!)

As well as Dr Kidman’s letter, in this issue we publish three more letters relating to the Journal’s editorial policies. It seems that a subject that was once too boring for polite company has suddenly become fashionable. Our earlier correspondent, Mr Jones of Hampton, has stirred the pot well. In the meantime, it is a matter of some irony for me that while the Journal has been under fire from his quarter for being too ‘academic’, it has simultaneously (believe it or not) been under fire from parts of the academic quarter for being ‘not academic enough’!

The crossfire reinforces my prejudice that the Journal’s recent policies have been reasonably close to the target. However I accept that there is rarely fire without heat, latent or otherwise. I have reached the view that Mr Jones may not be completely wrong after all, merely that he has selected from among the set of ACS publications the wrong target for his grape-shot. He reminds me of Goldilocks with only two bears: one is too heavy, and one is too light. He is really looking for a third that is just right!

It so happens that the ACS does have a third publication, that deserves more recognition than it gets — the proceedings of the annual Australian Computer Conference. This is not widely circulated except amongst conference attendees, and is most probably even less widely read. The packaging of the most recent version — it is about the size and weight of a junior house-brick — leaves something to be desired. It lists 64 papers and extends to 824 pages. The papers vary greatly in subject matter, presentation, erudition, intensity, length, formality, and reading difficulty. There is something here for everybody, even Mr Jones. The problem is to ensure that the useful part of this material can be distributed more widely, and in digestible increments.

I would like to propose to the ACS management that they consider seriously publishing what I shall call, for want of a better alternative, the Australian Journal of Computing Conference Papers. As I see it, this journal would reprint all the worthwhile conference papers, not only from the ACC but from many of the other related conferences now springing up around Australia. Papers would be selected democratically by an editorial board. Providing authors provide their manuscripts in electronic form, reformatting into a uniform style no longer need be difficult. (For example, three of the four papers in this Journal issue began life variously as a MacWrite file, as a Wordstar 2000 file, and as a UNIX troff file. The fourth, which you can most probably pick without difficulty, was typeset conventionally. Can you pick the others?) Modern computing and printing technology could make the production task for this new journal manageable and affordable without compromising much on quality. What would you think of that, Mr Jones?
Plausible Reasoning

J. L. O’Neill

A brief introduction to rule-based expert systems is followed by a review of the best-known approaches to reasoning with uncertainty, including subjective Bayesian reasoning, as typified by PROSPECTOR, measures of belief and disbelief, represented by MYCIN and its derivatives, and probability bounding methods, including the Dempster-Shafer theory of evidence and INFERNO. A small example is used to illustrate each inference method.

Keywords and Phrases: Bayes’ rule, belief measures, inference rules, INFERNO, MYCIN, PROSPECTOR, Dempster-Shafer theory

CR Categories: I.2.3 (Deduction)

INTRODUCTION

Rule-Based Expert Systems

The rule-based paradigm for representing knowledge has proven to be a powerful tool for capturing human decision-making expertise. The basic idea is that fragments of knowledge can be encapsulated in a set of inference rules of the form

situation — action

which can be interpreted as a conditional sentence meaning that when the situation is observed then the action should be taken. For this reason these rules are often referred to as situation-action rules.

The situation is usually restricted to conjunctions of conditions which may hold in some current data base consisting of assertions (assumed facts). A rule is said to be triggered when all the conditions in its situation hold. If the rule is fired (i.e. invoked) then the action typically modifies the data base to produce a new situation which may trigger other rules. The basic inference procedure which supports this successive application of rules is known as modus ponens. Potentially, many rules may be triggered by a given situation. The procedure which selects the most appropriate rule to fire, and fires it, is called an interpreter. Complex interpreters may use metarules to determine which of several competing rules to fire (i.e. to perform conflict resolution).

Rules are referred to variously as production rules, rewrite rules and antecedent-consequent rules in the special case where the system is constrained to deduce new facts directly from known facts. In this case the rules may be represented as the implication

antecedent — consequent

and the system is known as a simple deduction system.

A set of rules, a data base and an interpreter is known as a rule-based system. A rule-based system is a special case of a knowledge-based system in which the knowledge is encoded as rules. Where this knowledge in the rules has been derived from human experts, the term rule-based expert system is used.

Complex rule-based expert systems with several thousand rules have been constructed. For example DEC’s VAX configurer, XCON, also known as R1 (Bachant and McDermott, 1984), has over 4000 rules and is still evolving. The approach has become popular for several reasons: rules seem to correspond well to the way in which experts apply their knowledge; experts are able to express their knowledge in a form from which rules can be derived; and rules are simple structures, which makes it easy to build and modify complex knowledge bases, and provide stylised explanations for how and why rules were invoked. The technology also exists for the automatic acquisition of knowledge in the form of rules. (See for example, Michie, 1982, or Quinlan, 1983c.)

Control Strategies

As mentioned earlier, one of the roles of the interpreter is to determine which rules to fire and when to fire them. There are two basic strategies: data-driven and goal-driven. Data-driven control is also known as forward-chaining, bottom-up, pattern-directed and antecedent reasoning, while goal-driven control is referred to as backward-chaining, top-down, and consequent reasoning.

Data driven control repeatedly scans the data base to determine which rules have their antecedents satisfied, and selects a rule to be fired. Goal driven control selects a goal to be proved (i.e. some
Plausible Reasoning

The rules supplied by the expert and the assertions in the data base constitute the information available to the interpreter. This information will usually be uncertain, incomplete and inaccurate, and it is often difficult to make categorical statements about the truth or falsity of a set of alternatives. An expert may be willing to indicate which possibilities are the most likely, but 'the more specific he is required to provide the information the less confident he can be about it' (Yager, 1984). Strict logical implication gives way to weak implication or suggestion in rules derived from an expert, and 'facts' are often not known to be precisely true or false. Rules are therefore more appropriately interpreted as evidence → hypothesis

where we say the evidence suggests the hypothesis with some strength. Both the evidence and the hypothesis are uncertain propositions. In practice, a given piece of evidence will bear on more than one hypothesis, and a given hypothesis will be suggested by more than one piece of evidence. Therefore, rule sets may be represented by an inference network, in which propositions are represented as nodes with attached probabilistic validity measures, linked by weighted arcs to indicate the strength of an inference rule.

It is often impractical to use classical Boolean reasoning to draw conclusions from uncertain information. In the rare case when the information is sufficiently precise, it may be possible to use a logical reasoning tool, such as PROLOG, to construct an expert system directly (Clarke and McCabe, 1982, and Merry, 1983).

To handle the more usual case, a number of plausible reasoning strategies have been tried. Methods for plausible reasoning in knowledge-based expert systems (Hayes-Roth, 1984, or Hayes-Roth, Waterman and Lenat, 1983) require the assignment of some form of numerical validity measure to each fact or rule. The interpreter (also known as an inference engine) provides a control structure to propagate validity measures over inferences. Typically some assumptions are made regarding the independence of individual pieces of evidence, and in more recent approaches, attempts are made to detect and handle inconsistent (i.e. contradictory) propositions.

This paper reviews the best-known approaches to reasoning with uncertainty. These approaches include subjective Bayesian reasoning, as typified by PROSPECTOR (Duda, Hart, Barrett, Gaschnig, Konolige, Reboh and Slocum, 1978b), measures of belief and disbelief, represented by MYCIN and its derivatives (Buchanan and Shortliffe, 1984), and probability bounding methods, including the Dempster-Shafer theory of evidence (Shafer, 1976) and INFERNO (Quinlan, 1983a).

SUBJECTIVE BAYESIAN REASONING

Bayes' Rule

In probability theory, the objective probability P(C) is a measure of the frequency of occurrence of event C in a set of trials, but when P(C) is obtained by asking an expert to provide an estimate, which is really a degree of belief, it is called the subjective prior probability (Savage, 1971). This prior belief can be updated in the light of new evidence E, to the conditional probability P(C|E), possibly as the result of executing a production rule of the form E → C. This can be considered as a cause (C) and effect (E) relationship. However, a domain expert may be more able to estimate the probability of an effect given a cause, P(E|C).

Bayes' rule is a method of deriving P(C|E) from P(E|C). Given two events, A and B, the general multiplication rule of statistics tells us that the probability of both events occurring is the probability that one will occur times the probability that the other will occur, given that the first event has occurred. This can be expressed formally in two equivalent forms,

\[
P(A \& B) = P(B) \times P(A \mid B)
\]

\[
P(A \& B) = P(A) \times P(B \mid A)
\]

and by equating the two forms and rearranging, we get:

\[
P(B \mid A) = P(A \mid B) \times (P(B) / P(A))
\]

In terms of cause (C) and effect (E) above, we would have Bayes' rule as:

\[
P(C \mid E) = P(E \mid C) \times (P(C) / P(E))
\]

As an example of the application of Bayes' rule, suppose that 30% of all university entrants will fail [i.e. P(F_ Uni) = 0.3]. An hypothetical screening test based on HSC results correctly identifies 90% of all failures [i.e. P(F_scr | F_ Uni) = 0.9], but erroneously predicts that 5% of successful students will fail [i.e. P(F_scr | ~ F_ Uni) = 0.05]. What is the
probability that a student who has been predicted to fail will fail [i.e. \( P(F_{\text{Uni}}| F_{\text{Scr}}) \)]?

In Figure 1, we can get to \( F_{\text{Scr}} \) via two routes, hence \( P(F_{\text{Scr}}) = 0.27 + 0.035 = 0.3050 \). We can therefore compute

\[
P(F_{\text{Uni}}| F_{\text{Scr}}) = \frac{0.9 \times 0.3}{0.3050} = 0.8852.
\]

The use of Bayes' rule for expert systems poses some major problems. The necessity to provide prior probabilities for all propositions, will, in most application domains, force the expert to make subjective estimates with more precision than he is able. As all probabilistic information is stored as point values, there is no indication of the degree of uncertainty or even contradiction in the evidence which led to the probability estimates. In spite of these difficulties, successful expert systems have been based on Bayes' rule.

**PROSPECTOR**

PROSPECTOR (Duda et al., 1978b) is an expert system for mineral exploration which employs subjective Bayesian reasoning. It therefore requires the expert to provide prior probabilities for each proposition, assumes evidence to be conditionally independent and combines the pro and con evidence for a proposition into a point value representing probability, with the attendant lack of precision, and loss of information.

Rules in PROSPECTOR are represented as an inference network, and as is common with inference network systems, information is propagated in only one direction, from evidence to hypothesis.

**Likelihood Ratios**

Given an hypothesis \( H \) with prior probability \( P(H) \) and a rule \( E \rightarrow H \), then if \( E \) is true, Bayes' rule can be used to compute

\[
P(H|E) = P(E|H) \times \frac{P(H)}{P(E)} \quad \text{and}
\]

\[
P(\neg H|E) = P(E|\neg H) \times \frac{P(\neg H)}{P(E)}.
\]

Noting that odds and probabilities are related by \( O = P / (1 - P) \), and that \( \neg P = 1 - P \), dividing these two equations allows Bayes' rule to be written in odds-likelihood form as

\[
O(H|E) = \frac{LS}{O(H)}, \quad \text{where}
\]

\[
LS = P(E|H) / P(E|\neg H).
\]

LS is a **likelihood ratio** known as the **sufficiency measure**, so-called because if \( LS \) is infinite, then \( E \) is sufficient to establish \( H \). An analogous form for \( E \) absent is

\[
O(H|\neg E) = LN \times O(H), \quad \text{where}
\]

\[
LN = \frac{P(E|H)}{P(E|\neg H)}.
\]

LN is known as the **necessity measure**, because if LN is zero, \( E \) is necessary to establish \( H \) (Duda, Gaschnig and Hart, 1979). LS and LN may therefore be interpreted as rule strengths, which usually would be estimated by the expert. It would be nice if LS and LN were independent, but it can be shown that they are not (Duda, Hart and Nilsson, 1976). Hence the strengths supplied by the expert will almost certainly be inconsistent. In particular, if \( LS > 1 \), then \( LN < 1 \). This means that if \( E \) is confirming evidence for \( H \), then the absence of \( E \) must reduce the odds on \( H \). This does not, however, accord with the reasoning processes of experts.

**Conditional Independence**

Logarithms of likelihood ratios are sometimes considered to represent the strength or weight of a rule. If there is a set of rules which imply \( H \), (say \( E_i \rightarrow H, \ i = 1, \ldots, n \)), then to use Bayes' rule, we would let \( E \) be the conjunction of the \( E_i \)'s, and attempt to compute the likelihood ratio

\[
L = \frac{P(E_1 \& E_2 \& \ldots \& E_n | H)}{P(E_1 \& E_2 \& \ldots \& E_n | \neg H)}.
\]

This requires that we know the weights for all the possible combinations of the \( E_i \)'s (in pairs, triples, \ldots). As this is impractical even for small \( n \), the simplifying assumption is made that the \( E_i \) are **conditionally independent**. That is to say that for two pieces of evidence \( E_i \) and \( E_j \),

\[
P(E_i \& E_j | H) = P(E_i | H), \quad \text{and}
\]

\[
P(E_i | E_j \& \neg H) = P(E_i | \neg H)
\]

so that in general

\[
P(E_1 \& E_2 \& \ldots \& E_n | H) = \prod_i P(E_i | H).
\]

This allows \( L \) to reduce to the product of the individual \( L_i \)'s, and we can calculate \( P(H|E_1 \& E_2 \ldots \& E_n) \). Unfortunately, this assumption is unreasonable, and must be violated for any inference network which is not a tree, as at least some of the \( E_i \) would have a common ancestor. To minimise the problems which result from making the conditional independence assumption, the pragmatic knowledge engineer must design the rule set to keep the number of premises per rule, and the number of rules that conclude \( H \), as small as possible (Buchanan and Duda, 1983).

In accordance with the terminology of Duda et al., we will refer to evidence and hypotheses as assertions when it is convenient. PROSPECTOR uses **logical relations**, **plausible relations** and **contextual relations** to control the way a change in probability of one
Plausible Reasoning

assertion will affect the probability of other assertions.

Logical Relations
For logical relations, the validity (truth or falsity) of an hypothesis, $H$, is completely determined by the validity of its defining assertions, using Zadeh's fuzzy-set formulae. Therefore, if the validity of at least one of the defining assertions cannot be determined, then the probability of $H$ may remain unchanged. If this is undesirable, then plausible relations may be used.

Plausible Relations
For plausible relations, assertions are combined using the odds-likelihood form of Bayes' rule, with modifications. Bayes' rule can only be used where the evidence $E$ (or $\neg E$) is certain. In practice, $E$ may be uncertain, because either $E$ was declared by a user to be uncertain, or $E$ was deduced from another plausible relation, using evidence $E'$ (say), yielding $P(E|E')$. The problem of computing $P(H|E)$ becomes one of computing $P(H|E')$, which can be shown (Duda et al., 1976) to be calculable (with assumptions), from

$$P(H|E') = \frac{P(H|E) \times P(E|E')}{P(H|\neg E) \times [1 - P(E|E')]}.$$

If $E$ ($\neg E$) is known with certainty, this formula produces consistent results. However, if $E'$ is irrelevant to $E$, then $P(E|E') = P(E)$, and the formula should produce a value for $P(H|E')$ which agrees with the expert's estimate of the prior probability $P(H)$. This is unlikely, leading to the conclusion that $P(H)$, $P(E)$, $P(H|E)$ and $P(H|\neg E)$ are not independent.

To solve this problem, PROSPECTOR uses a piece-wise linear function of $P(E|E')$ to compute $P(H|E)$ for each rule, with a way-point to ensure that $P(H|E') = P(H)$ supplied by the expert. This is illustrated in Figure 2 (Duda et al., 1978a). Converting to odds yields $O(H|E')$, and hence an effective likelihood ratio

$$L = \frac{O(H|E')}{O(H)}$$

can be computed for each rule. This ratio is dynamic, tending towards LS as $E$ is supported, and towards LN as $E$ is refuted. If $n$ rules determine $H$, each with effective likelihood ratio $L_i$, the conditional independence assumption allows posterior odds on $H$ to become

$$O(H|E') = O(H) \times L_1 \times L_2 \ldots \times L_n.$$

This forms the basic propagation mechanism used by PROSPECTOR, and has the advantage that the same result will be produced irrespective of the order in which evidence is considered.

Contextual Relations
Contextual relations inhibit the use of an assertion until some other assertion has been established. This ensures that questions are not asked about objects which may not exist.

An Example
To illustrate the operation of the PROSPECTOR propagation mechanism, the fragment of an inference network in Figure 3 considers some of the factors which may be relevant to predicting whether a given candidate would make a good computer science student.

The nodes Analytic and Good CS are hypotheses and the leaf nodes 4-Unit Math, HSC $> 400$ and Hobbyist are observable evidence. Each hypothesis has a prior probability. The numbers on the arcs are the sufficiency and necessity likelihood ratios respectively. The model suggests that 4-Unit mathematics and high HSC scores are good indicators of analytic ability, with mathematics the better metric ($LS = 100$). An analytical mind is more suggestive of good CS prospects than being a home computer hobbyist. The logical OR node ensures that a candidate with neither 4-Unit Math nor a good HSC score will not be considered to have good analytic ability.

Suppose that a given student has certainly completed 4-Unit Math. We can now apply PROSPECTOR's version of Bayes' rule to compute a posterior probability of the candidate being a Good CS. We can convert the prior probability of Analytic to odds $[0.05/(1-0.05) = 0.0526]$, multiply by LS [100], and convert back to probability, to get a posterior probability of Analytic $[5.26/(1+5.26) = 0.84]$. 

The nodes Analytic and Good CS are hypotheses and the leaf nodes 4-Unit Math, HSC $> 400$ and Hobbyist are observable evidence. Each hypothesis has a prior probability. The numbers on the arcs are the sufficiency and necessity likelihood ratios respectively. The model suggests that 4-Unit mathematics and high HSC scores are good indicators of analytic ability, with mathematics the better metric ($LS = 100$). An analytical mind is more suggestive of good CS prospects than being a home computer hobbyist. The logical OR node ensures that a candidate with neither 4-Unit Math nor a good HSC score will not be considered to have good analytic ability.

Suppose that a given student has certainly completed 4-Unit Math. We can now apply PROSPECTOR's version of Bayes' rule to compute a posterior probability of the candidate being a Good CS. We can convert the prior probability of Analytic to odds $[0.05/(1-0.05) = 0.0526]$, multiply by LS [100], and convert back to probability, to get a posterior probability of Analytic $[5.26/(1+5.26) = 0.84]$. 

Figure 3.
Plausible Reasoning

Propagating this result, we can revise the effective strength of the rule from Analytic to Good CS by interpolation \([0.84-0.05]/(1-0.05) \times 200 = 166.32\) from the increase in the probability of Analytic from its prior value. This enables us to calculate a posterior probability of being a Good CS \([\text{via odds on } 0.01/(1-0.01) \times 166.32 = 1.68]\) of 0.63. PROSPECTOR maps probabilities onto a [-5, 5] subjective certainty scale for the user’s benefit, where the certainty is intended to reflect the change in probability, from its prior value. The certainty of being a Good CS becomes 5 \times (0.63-0.01)/(1-0.01) = 3.13.

Rule Ordering

In a goal-directed expert system we are interested in all rules which could imply the current goal. Potentially, there will be many of these, and therefore some mechanism must be used to select the most appropriate rule. Initially, PROSPECTOR selects the top-level hypothesis, H, that has the highest current probability. It then uses an empirical ranking function (Duda et al., 1978a)

\[
\text{abs log}(L / LN) \times [P(E|E') + \text{abs log}(L / L)] \times [1 - P(E|E')]
\]

where L is the effective likelihood ratio, to select the highest scoring untried rule which concludes H, and if its antecedent is askable, any information the user can provide is propagated and the ranking procedure is repeated. For non-askable antecedents, or if the user cannot help, the procedure is invoked recursively with the antecedent as the hypothesis. The ranking function is a heuristic which favours necessary conditions when E is improbable, and sufficient conditions as E becomes more likely, although if E (¬E) is well determined, low scores result, to inhibit unproductive rules. Duda et al. concede that the development of an optimal rule ordering procedure is infeasible for large networks.

Summary

PROSPECTOR’s plausible reasoning strategy, as an attempt to apply Bayes’ rule, has a reassuring basis in probability theory. However, there are problems with this approach which PROSPECTOR does not adequately solve. In particular, we have seen that the prior probabilities and likelihood ratios supplied by the domain expert will inevitably be inconsistent, and there is no theoretical basis for the piece-wise linear ‘fiddle’ adopted to hide the problem. Prior probabilities are not always able to be estimated, although in the geological domain, sufficient statistical data has been collected over the years to provide reasonable approximations. A further problem arises when the user realises that previously offered evidence should be retracted. How are the probabilities of all dependent hypotheses to be altered? PROSPECTOR’s treatment of contradictory evidence is unsatisfactory for problems which are characterised by highly conflicting data and opinions. The merging of all evidence into a single probabilistic measure hides conflict and tends to overstate the precision with which conclusions can be drawn. Finally, the assumption of conditional independence in the evidence is unjustifiable, but this is a problem which has yet to be completely solved by any plausible inference technique.

MEASURES OF BELIEF AND DISBELIEF (MYCIN)

Introduction

MYCIN (Buchanan and Shortliffe, 1984; Davis, Buchanan and Shortliffe, 1977), is a Lisp-based consultation system developed at Stanford in the mid-1970’s to provide diagnosis and antimicrobial therapy advice to physicians in relation to infectious diseases. It uses a backward-chaining (goal-driven) search strategy, whereby the diagnosis problem is established as a goal, and sub-goals are recursively created from the premises of any rule which can infer the goal.

The use of Bayes’ theorem as a statistical procedure for modifying a priori probabilities in the light of further evidence would, in the medical domain, require comprehensive statistics relating symptoms to diseases. Such data would generally be either unobtainable or inaccurate. Personal probabilities, as in PROSPECTOR, would not be a viable alternative, as experts seem unable to provide consistent prior probabilities. In medical diagnosis, however, it is not critical to determine precise probabilities for the presence of a disease-causing organism. It is sufficient to identify the few most likely causes, and to treat them. In other words, the objective is a sound diagnosis. Therefore, the MYCIN developers have devised an empirical method for the estimation of the likelihood of a disease, given a serially supplied set of symptoms.

Belief Measures

The reader will recall from the discussion of PROSPECTOR, that the Bayesian approach would suggest that the absence of confirming evidence for an hypothesis should reduce the likelihood of the hypothesis. MYCIN recognises that there are many practical instances where this does not hold. Two measures are introduced, in order to separate confirming and disconfirming evidence. MYCIN records measures of belief and disbelief in a hypothesis, given the evidence, in recognition that human experts work with beliefs, rather than conditional probabilities. Using MYCIN notation:

\[
\text{MB}[H, E] = x
\]

means ‘the measure of increased belief in the hypothesis H, based on the evidence E, is x’, and

\[
\text{MD}[H, E] = y
\]

means ‘the measure of increased disbelief in the hypothesis H, based on the evidence E, is y’, (Shortliffe and Buchanan, 1975), where evidence may itself be an hypothesis.
Therefore, evidence does not have to be certain, but may have its own measures of belief and disbelief. It would probably be better to express MB and MD respectively in terms of E_i and E_a, being the evidence for and against the hypothesis. MB and MD are not probabilities. Rather, initial values are determined from physicians' estimates of the extent to which evidence tends to confirm or disconfirm the hypothesis. It is, however, possible to interpret them in terms of conditional probability. If we had a prior belief in an hypothesis of P(H), then we could say that we had a prior disbelief of 1-P(H). If some observation E supported H, then, provided H was not certain (i.e. P(H) < 1), we would tend to believe H more, or by implication, disbelieve H less.

Proportionately how much less would be given by the ratio:

\[
\frac{P(H|E) - P(H)}{1 - P(H)}
\]

which MYCIN represents as MB[H, E]. A similar argument for the case where E is discouraging for H leads to an interpretation of MD[H, E]:

\[
\frac{P(H) - P(H|E)}{P(H)}
\]

From these probabilistic interpretations it can be seen that if E does not alter P(H|E) from its prior value P(H), then both MB[H, E] and MD[H, E] would be zero, reflecting the irrelevance of E to H. Furthermore, if E makes H certain, then MB[H, E] = 1, or if E rules out H (i.e. P(H|E) = 0), then MD[H, E] = 1. Therefore, both MB and MD are constrained (like probabilities) to lie in the interval [0, 1], but belief measures should not be viewed as placing bounds on the probability of a hypothesis. MYCIN makes the assumption that the expert's degree of belief is consistent with the values of MB and MD that would be calculable from the above formulae in the event that the prior and conditional probabilities were known. Because they are point values, belief measures are subject to the usual objections about precision, such as 'A likelihood represented by a point value is usually an overstatement of what is actually known' (Garvey, Lowrance and Fischler, 1981).

An important characteristic of belief measures is that any evidence which confirms an hypothesis to some degree must also disconfirm the negation of the hypothesis to the same degree. That is,

\[
MB[H, E] = MD[-H, E].
\]

It would be nice if, for a given hypothesis, the MB and MD for each piece of evidence could be independently solicited from the physician. Unfortunately, this is not the case, and so, as in PROSPECTOR, internal consistency problems are inevitable. If two pieces of evidence relevant to a diagnosis were independent, then we would expect that (Adams, 1976):

\[
P(H|E_1) \times P(H|E_2) = P(H|E_1 \& E_2) \times P(H)
\]

If we were told that symptom E_1 is conclusive evidence for disease H (i.e. P(H|E_1) = 1), then we would be forced to the unpalatable conclusion that P(H|E_2) = P(H), and hence that symptom E_2 is irrelevant to H! Obviously, this would be unlikely to correspond to the intent of the diagnostician.

**Certainty Factors**

In order to weigh the evidence for and against a hypothesis, and to rank alternative hypotheses, MYCIN introduces the concept of a certainty factor, defined simply as

\[
\]

and hence CF lies in the interval [-1, 1]. CF is clearly not a probability, and in fact examples have been constructed (Adams, 1976) that show two hypotheses, where the one with the lower CF has the higher probability! However, for the case of confirming evidence E, if the prior probability P(H) is very low, substitution in the probabilistic formulae for belief measures shows that CF[H, E] can approximate P(H|E). The use of CF has been defended on the grounds that it is an intuitive concept that corresponds well to the way experts balance evidence, although it is still necessary to retain the separate MBs and MDs, so that, inter alia, the order in which evidence is considered does not affect the result.

EMYCIN — for Essential MYCIN — (Van Melle, 1980), is an expert system shell based on MYCIN. A shell is a domain independent inference engine, incorporating a reasoning mechanism, knowledge acquisition tools and explanation facilities, but devoid of a knowledge base. The intent is to allow the construction of an expert system through the addition of problem-specific knowledge. EMYCIN discards belief measures, propagating only the certainty factor, and so the strength of the confirming and disconfirming evidence is lost. In the absence of MB and MD it is not possible to determine whether a CF of zero implies a lack of any knowledge or equal (non-zero) values of MB and MD. EMYCIN's only allowance for this is not to assert the conclusion of a rule unless the CF of its antecedent is greater than 0.2. This ad hoc approach has the advantage that it tends to restrict the triggering of less effective rules, but it means that the CF of a conclusion is not a strictly continuous function of the certainty of the evidence (Buchanan and Duda, 1983). Once the reasons for believing (disbelieving) an inference have been lost, it is impossible for any explanation system to justify why a conclusion is believed (disbelieved). The explanation would be limited to a trace of how the conclusion was reached (Cohen and Grinberg, 1983).

It can be shown (Shortliffe, 1974, 1976) that for any number of mutually exclusive hypotheses confirmed by observation E, that

\[
\sum_i (CF[H_i, E]) \leq 1
\]
This requirement can be used to ensure that the CFs supplied by the expert are consistent.

Combining Functions
Belief measures are only useful if they can be combined in some way, and in MYCIN, if symptom S2 is observed after symptom S1, then this can be used to update the belief in hypothesis H, using the following rules (Shortliffe and Buchanan, 1975):

1. Incrementally acquired evidence:
   \[ MB[H, S_1 \& S_2] = \begin{cases} 1 & \text{if } MB[H, S_1] = 1 \\ MB[H, S_1] + MB[H, S_2] (1 - MB[H, S_1]) & \text{else} \end{cases} \]
   \[ MD[H, S_1 \& S_2] = \begin{cases} 0 & \text{if } MB[H, S_1] = 1 \\ MD[H, S_1] + MD[H, S_2] (1 - MD[H, S_1]) & \text{else} \end{cases} \]

The formulae for incrementally acquired evidence are analogous to the probability formula for combining independent events, and because new evidence can only result in increased values of MB and MD, these formulae would not be useful in all problem domains. The success of MYCIN suggests that, at least in medical diagnosis, successive clinical observations tend not to contradict each other.

To determine the belief in the conjunction or disjunction of two hypotheses, in the light of the evidence E, which is the total of all the clinical observations to date, MYCIN uses formulae which resemble the fuzzy formulae of Zadeh (1979a):

2. Conjunctions of hypotheses:
   \[ MB[H_1 \& H_2, E] = \min (MB[H_1, E], MB[H_2, E]) \]
   \[ MD[H_1 \& H_2, E] = \max (MD[H_1, E], MD[H_2, E]) \]

3. Disjunctions of hypotheses:
   \[ MB[H_1 | H_2, E] = \max (MB[H_1, E], MB[H_2, E]) \]
   \[ MD[H_1 | H_2, E] = \min (MD[H_1, E], MD[H_2, E]) \]

The use of the fuzzy formulae has been noted (Quinlan, 1983a) to be inconsistent in that the resultant measure of belief for conjunctions is very optimistic, while for disjunctions it is overly pessimistic. In particular, if H1 and H2 were mutually exclusive, then we would expect the disjunction of H1 and H2 to be more likely than merely the most likely of the two. Quinlan has also observed that MYCIN would have difficulty in identifying inconsistency, except in the rare case where both MB and MD were unity. For any other values of the belief measures, it is unclear whether (MB + MD > 1) is a contradiction, because belief measures are not probabilities. Adams (1976) has noted that there are ‘strong restrictive assumptions’ on H1 and H2. For example, if the hypotheses are mutually exclusive, then the conjunction of H1 & H2 must be false, irrespective of any evidence.

When an expert has been asked to supply a certainty factor for a rule, he will assign a CF (via MB'[H, S1] and MD'[H, S1]) based on the assumption that the rule’s premises are true. It will often be the case that the truth of the premises cannot be known with certainty, and may themselves be subject to confirmation in the light of available evidence. In other words, if S1 is an hypothesis these formulae can be used to adjust the belief measures, and hence CF:

4. Strength of evidence
   \[ MB[H, S_1] = MB'[H, S_1] \times \max (0, CF[S_1], E) \]
   \[ MD[H, S_1] = MD'[H, S_1] \times \max (0, CF[S_1], E) \]

In order that the user does not have to supply categorical replies to MYCIN’s questions CFs are used in the evaluation of all information.

An Example
As an illustration of the application of the MYCIN combining functions, consider again our problem of predicting whether a given student will be a good Computer Science student.

Suppose we have another rule, R, that says that CF[H, S1&S2&S3] = 0.7, where, as usual, the expert has assumed S1, S2 and S3 to be certain. If S2 and S3
were known to be certain, but $S_1$ was uncertain, with $CF[S_1, E] = 0.6$, then the strength of $R$ would have to be reduced. Using the 'strength of evidence' combining function:

$$MB[H, S_1, S_2, S_3] = MB[H, S_1, S_2, S_3, E] \times \max (0, CF[S_1, S_2, S_3, E]).$$

This requires the calculation of $CF[S_1, S_2, S_3, E]$, the certainty of which can be calculated from the 'conjunction of hypotheses' combining function to be 0.6. We would then have:

$$MB[H, S_1, S_2, S_3] = 0.7 \times \max (0, 0.6) = 0.42,$$

which would be used to update $MB[H, E]$ via the combining function for 'incrementally acquired evidence'.

The Final Diagnosis
Because CFs are not probabilities, and there is a need to ensure that all likely diseases are treated, MYCIN does not merely select the hypothesis with the highest CF when no more rules will fire. The competing hypotheses are ranked, $\sum_i (CF_i)$ is calculated, and organisms corresponding to the highest ranking hypotheses that account for 90% of the total are selected for treatment. This approach can compensate for a multitude of sins in the propagation process. Although hypotheses may not rank as the system would under a Bayesian regime with accurate statistics, in practice there is a fair chance of all likely hypotheses being noted.

Rule Ordering
In contrast to PROSPECTOR’s ranking function, MYCIN uses meta-rules to select the most useful relevant rule to apply (Davis, 1976). Meta-rules are rules which reason about rules, and contain strategies for selecting the most promising chain of reasoning. This approach is appealing because these control rules use the same representation as the lower level knowledge, and are therefore as easy to modify. The potential also exists for the system to explain, in a more comprehensible fashion, why it selected a particular line of reasoning.

Because it is important in medical diagnosis to consider all possible diseases, MYCIN’s general strategy is to apply all relevant rules at each step (i.e. all rules in the conflict set). However, because the conflict resolution algorithm is unheaded from the interpreter, there is the possibility of applying goal-dependent rule selection strategies.

As Davis points out, the prospect of even higher level meta-rules exists. For example, it may be useful to have rules which are responsible for selecting the strategy rules, and even higher level rules for determining what criteria are most important to assess in determining a strategy.

Summary
MYCIN’s approach to plausible reasoning, represents a departure from the more probabilistic methods of PROSPECTOR. Certainty factors have no reasonable probabilistic interpretation, and as in PROSPECTOR, the balance of evidence can be disguised through the use of such a single measure. The elimination of the need to estimate prior probabilities is an important advance. However, the invalid assumption that evidence is independent can make it impossible for the expert to assign consistent rule strengths.

Shortliffe and Buchanan acknowledge that their methods have some shortcomings, but at least they have been demonstrated to allow expert judgmental knowledge to be used in medical diagnosis when the statistical data needed to use Bayes’ Theorem are not available. They claim that ‘the certainty factor approach is robust enough for use in a decision-making domain such as antimicrobial selection.’ It has been suggested (Adams, 1976) that this success has been due to the fact that MYCIN uses short reasoning chains and simple hypotheses, an approach which may be infeasible in other domains.

DEMPSTER-SHAFER THEORY OF EVIDENCE
Introduction
The Dempster-Shafer theory is an attempt to deal with incomplete and uncertain evidence, and provide an estimate of the degree of conflict in the evidence. The theory, which was conceived by Dempster (1968), and formulated by his student, Shafer (1976), reduces to Bayesian reasoning when knowledge is exact.

Unlike the certainty factor (CF) approach of MYCIN, in which a given piece of evidence bears on a single hypothesis, evidence is allowed to affect any subset of the set of all hypotheses. This set of hypotheses may be viewed as just another hypothesis with some likelihood, with the advantage that the evidence need not assign beliefs to the individual hypotheses of the subset. In other words, evidence is not required to be more specific than it is able. This is an improvement over Bayesian belief functions, where belief must be assigned to each of the singleton hypotheses, even when the evidence is insufficiently detailed.

The theory assumes a fixed set of mutually exclusive and exhaustive environmental possibilities, $\theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$, the powerset of which is known as the frame of discernment. This powerset can be represented as a lattice. For example, if we assume the set of possibilities $\theta = \{A, B, C, D\}$, then the frame of discernment (Strat, 1984) is as shown in Figure 5.

This powerset has $2^n$ subsets, including the null set, which is not shown in Figure 5. The requirement for the set of possibilities to be exhaustive ensures that the null set must represent an hypothesis which is false. Subsets of $\theta$ are propositions in the domain.
For example, when rolling a die, there are six possibilities for the resultant number, and the proposition that the number is even corresponds to the subset of $\{1, 2, 3, 4, 5, 6\}$ that asserts that the die shows two, four or six (Barrett, 1981). This formalism "translates the logical notions of conjunction, disjunction, implication and negation into the more graphic, set-theoretic notions of intersection, union, inclusion, and complementation" (Garvey, Lowrance and Fischler, 1981).

**Belief Functions**

Evidence is represented by a mass distribution, $m$, of a unit of belief across the set of propositions of $\{1, 2, 3, 4, 5, 6\}$. In functional form,

$$m : \{A_i | A_i \text{ is a subset of } \{1, 2, 3, 4, 5, 6\}\} \rightarrow [0, 1],$$

$$m(\text{null}) = 0, \quad \sum_i A_i = 1.$$

If the use of the term mass distribution seems a little strange, then it is useful to think of it as a basic probability assignment. The assignment of some fraction of the total mass to a proposition, say $H$, represents partial commitment to $H$ and all propositions $H$ implies, while remaining neutral to propositions that imply $H$ (i.e. descendant nodes) (Strat, 1984). The belief assigned to $H$ cannot be apportioned to the individual subsets of $H$. The propositions to which mass has been assigned are referred to as the focal elements of $m$. It is an important feature of the mass distribution that the focal elements need not be mutually exclusive. The Dempster-Shafer theory defines a belief function, the support for $H$, as the total mass assigned by $m$ to $H$ and its descendants:

$$\text{Spt}(H) = \sum_i H_i, \text{ for } H_i \text{ a subset of } H.$$

The plausibility of $H$ is the mass assigned to propositions that do not imply $\sim H$:

$$\text{Pls}(H) = 1 - \text{Spt}(\sim H).$$

[Spt($H$), Pls($H$)] thus defines an evidential interval, where the residual uncertainty in $H$ is given by Pls($H$) - Spt($H$), which is sometimes referred to as the ignorance of $H$, Igr($H$) (Wesley, Lowrance and Garvey, 1984), and corresponds to the mass assigned to supersets of $H$. The reasoning involved in the calculation of support and plausibility is referred to as extrapolation (Lowrance and Garvey, 1983) where belief in a proposition may involve belief in some other propositions. As the mass distribution (or basic probability assignment) assigns a unit of belief, then by definition Spt($\emptyset$) = 1. Note that unlike the Bayesian approach, in which the balance of the belief in an hypothesis is assigned to its negation, there is no assumption that $P(H) = 1 - P(\sim H)$. However, in the special case where Igr($H$) = 0, then the evidential interval merges to become a point probability, and the Bayesian method becomes a special case of the Shafer representation.

The following examples of evidential intervals and their interpretations is from Garvey et al., 1981:

- $H[0, 1] \rightarrow$ no knowledge at all about $H$.
- $H[0, 0] \rightarrow$ $H$ is false.
- $H[1, 1] \rightarrow$ $H$ is true.
- $H[.25, 1] \rightarrow$ evidence partially supports $H$.
- $H[0, .85] \rightarrow$ evidence partially supports $\sim H$.
- $H[.25, .85] \rightarrow$ evidence provides support for both $H$ and $\sim H$.

**Combining Evidence**

Dempster's Rule of Combination may be used to pool evidence from two mass distributions, $m_1$ and $m_2$, representing the opinions of two knowledge sources, to produce a consensus, which Dempster refers to as the orthogonal sum:

$$m_3(P) = (1 - \kappa)^{-1} \sum (m_1(A_i) \times m_2(B_j)),\quad \text{for } (A_i \cap B_j) = P,$$

$$\kappa = \sum m_1(A_i) \times m_2(B_j),\quad \text{for } (A_i \cap B_j) = \text{null}.$$

To ensure that no mass is given to null, $m_3$ is normalised by the scaling factor $(1 - \kappa)^{-1}$, where $\kappa$ is the mass that is assigned to null. If $\kappa = 1$, the evidence represented by $m_1$ and $m_2$ is totally contradictory and no orthogonal sum is defined. Lesser values of $\kappa$ are an intuitive measure of the degree of conflict in the evidence, but normalisation tends to hide it. The weight of conflict is measured by $\log(\kappa)$. For a discussion of the validity of this process, see Zadeh (1979b). In graphical terms, mass can be assigned to propositions represented as sets of points on the number line, and the pooling of disparate sources can be viewed as the computation of set intersections within a unit square. For example, belief proportional to the shaded area in Figure 6 (from Barnett, 1981) would be committed to the proposition B & Y.

A given proposition may be associated with more than one rectangle, so the areas of all such rectangles would be summed. Any rectangle committed to null would be discarded, and the size of the remaining rectangles would be increased in proportion. This is only possible if there is at least one rectangle which is not committed to null. That is, provided the

---

*The Australian Computer Journal, Vol. 19, No. 1, February 1987*
Plausible Reasoning

UNIT SQUARE

<table>
<thead>
<tr>
<th>M1 (A)</th>
<th>M2 (X)</th>
<th>M2 (Y)</th>
<th>M2 (Z)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.

The evidence represented by the two mass assignments are not totally contradictory.

In applying Dempster's rule, some predictable rules must be observed: the conjunction of a proposition, H say, with itself (or with θ) is H; the conjunction of H with some other mutually exclusive proposition is null; and the conjunction of H with an overlapping proposition, P say, is H & P.

Because Dempster's Rule is associative and commutative, multiple bodies of evidence can be combined in any arbitrary order, to produce the same result. Recall that this was one of the objectives of the MYCIN scheme. However, if normalisation is performed after each pair of opinions have been pooled, then the total degree of conflict between all the knowledge sources will be lost. Catlett (1984) has been able to show that deferring normalisation until the final orthogonal sum is computed will produce the same results while extracting the total degree of conflict. A problem which still remains is how to interpret this conflict.

Bayesian conditional probability is a special case of the orthogonal sum, as, if the two belief functions to be pooled are Bayesian, then the correct conditional probabilities result. The Dempster-Shafer method has been adopted for applications where the statistical data required to use Bayesian reasoning is unobtainable, such as in the integration of data from multiple radar sensors (Lowrance and Garvey, 1983, and Garvey and Lowrance, 1984a, 1984b).

An Example
Suppose that, in terms of our computer science aptitude example, we are interested in whether a given prospect is likely to be a superior, good, average, marginal, poor, or unsuitable student. If these were the only possibilities, then θ would be:

θ = { Supr, Good, Avge, Marg, Poor, Unsb }

Following an interview, in which the prospect was scored on a number of attributes by the individual members of a panel, the interviewers were asked to express their opinions as to the quality of the candidate. The assessments were expressed as mass assignments. If the first panel member gave the favourable assessment:

\[ m_1(<\text{Supr, Good or Avge, not Unsb}, \theta>) = \begin{align*}
< 0.2, 0.1, 0.4, 0.3 >
\end{align*} \]

then the following evidential intervals could be calculated, using the definitions for support and plausibility. Recall that \( \text{Spt}(H) \) is the sum of the focal elements that imply H, and \( \text{Pls}(H) \) is \( 1 - \text{Spt}(\neg H) \):

\[ \begin{align*}
\text{Supr}[0, 1 - (0.1) = 0.9], \\
\text{Good}[0, 1 - (0.2) = 0.8], \\
\text{Avge}[0, 1 - (0.2 + 0.1) = 0.7], \\
\text{Marg}[0, 1 - (0.2 + 0.1 + 0.4) = 0.3].
\end{align*} \]

Note that the mass assignment to the proposition (Good or Avge) does not lend support to the respective singleton propositions. If the second panel member also gave a favourable opinion:

\[ m_2(<\text{Supr or Good, Avge,} \ θ>) = \begin{align*}
< 0.5, 0.2, 0.3 >
\end{align*} \]

then a consensus could be obtained, by applying Dempster's rule to compute the orthogonal sum in Figure 7.

Noting that some mass (0.04) has been assigned to null, it is necessary to normalise the result by scaling up all the other masses. The mass assignments which result from this are:

\[ \begin{align*}
\text{Proposition} & \quad \text{Mass} & \quad \text{Norm} \\
\text{Supr} & 0.16 & 0.17 \\
\text{Good} & 0.05 & 0.05 \\
\text{Avge} & 0.16 & 0.17 \\
\neg \text{Unsb} & 0.12 & 0.13 \\
\text{Supr} \lor \text{Good} & 0.35 & 0.36 \\
\text{Good} \lor \text{Avge} & 0.03 & 0.03 \\
\end{align*} \]

from which the evidential intervals for the possibilities can be calculated:

\[ \begin{align*}
\text{Supr}[0.16, 1 - (0.05 + .17 + .03) = .75], \\
\text{Good}[0.05, 1 - (.17 + .17) = .66], \\
\text{Avge}[0.16, 1 - (.17 + .05 + .36) = .42], \\
\neg \text{Unsb}[0.12, 1 - (.17 + .05 + .36 + .03) = .22], \\
\text{Supr} \lor \text{Good}[0.35, 1 - (.17 + .05 + .17 + .36 + .03) = .22], \\
\neg \text{Unsb}[0.12, 1 - (.17 + .05 + .17 + .36 + .03) = .22].
\end{align*} \]
Plausible Reasoning

It would appear that another interviewer will be required to provide a definitive assessment of this candidate!

Summary
One of the principle objections to the Bayesian approach, that uncertainty cannot be properly represented, is overcome by the Dempster-Shafer method. Bayesian theory would demand that whatever residual belief was not assigned to a proposition, must be assigned to its negation. In the Shafer representation it is quite permissible for a knowledge source to admit that it does not have sufficient information to assign belief to every proposition. The Bayesian requirement that the propositions to which belief is assigned (i.e. the focal elements) be mutually exclusive has also been eliminated.

The Dempster-Shafer approach separates the extrapolation of belief (from belief in a proposition to belief in others) from the pooling of separate bodies of evidence to produce a balanced consensus, unlike MYCIN and PROSPECTOR, which use only a single line of reasoning. However, the orthogonal sum operation still assumes that the evidence to be pooled is independent.

INFERNO

Introduction
INFERNO (Quinlan, 1983a) is a two-valued probabilistic reasoning system which does not make the usual conditional independence assumptions, so that, as Quinlan states, 'the correctness of inferences can be guaranteed'. Of course, false information will still lead to erroneous conclusions. It is the first system to enable the user to declare subsets of propositions to be independent. INFERNO is also able to detect any inconsistencies which may be present in propositions and suggest rectifications for them. All propositions are created equal, as INFERNO does not restrict the propagation of information to a single direction, from evidence to hypothesis, as is found in other systems.

Any proposition, A, is assigned two probabilistic values, t(A) and f(A), where t(A) is a lower bound on the truth of A, and f(A) is a lower bound on the falsity of A, in the light of the evidence. This representation overcomes the principal criticism of the single-valued subjective Bayesian approach, that 'there is trouble representing what we know without being forced to overcommit when we are ignorant' (Barnett, 1981). Provided that t(A) + f(A) ≤ 1 then the bounds and the evidence are consistent, and in terms of the Shafer representation Spt(A) = t(A), and Pls(A) = 1 - f(A). The bounds t(A) and f(A) are initially zero, and increase as information is gained, thereby increasing the precision with which the probability of A is known.

Inference Rules
INFERNO uses relations between propositions, and propagation constraints derived from them to compute higher values of the bounds. For example, Quinlan (1983a) gives the following relations (where the reference numbers for the formulae are the same as he uses):

A enables S with strength X
is interpreted as the weak implication 'P(S|A) ≥ X', from which the propagation constraints can be derived

\[
(1.1) \quad t(S) \geq t(A) \times X \\
(1.2) \quad f(A) \geq 1 - \frac{(1 - f(S))}{X}.
\]

A conjoins \{S_1, S_2, \ldots, S_n\}
is interpreted as

\[
'A = \bigcap_i S_i',
\]
from which the propagation constraints can be derived

\[
(3.1.1) \quad t(A) \geq 1 - \sum_i (1 - t(S_i)) \\
(3.1.2) \quad f(A) \geq f(S_i) \\
(3.1.3) \quad t(S_i) \geq t(A) \\
(3.1.4) \quad f(S_i) \geq f(A) - \sum_{j \neq i} (1 - t(S_j))
\]

Hence, for enables, whenever t(A) increases, t(S) may need to be increased, and whenever f(S) increases, f(A) must be checked. The form of constraints for other relations are similar, and are effectively a set of production rules for increasing the left hand bound whenever the expression on the right hand side increases. Mutual exclusion is also supported, and propositions may be declared to be independent, for example:

A disjoins-independent \{S_1, S_2, \ldots, S_n\}
is interpreted as

\[
'A = \bigcup_i S_i, \text{ and}
\]
for all \(i \neq j\), \(P(S_i \& S_j) = P(S_i) \times P(S_j)'.

The relevant propagation constraints are

\[
(4.2.1) \quad t(A) \geq 1 - \prod_i (1 - t(S_i)) \\
(4.2.2) \quad f(A) \geq \prod_i f(S_i) \\
(4.2.3) \quad t(S_i) \geq 1 - (1 - t(A)) / \prod_{j \neq i} f(S_j) \\
(4.2.4) \quad f(S_i) \geq f(A) / \prod_{j \neq i} (1 - t(S_j))
\]

A complete list of these relations and derivation of the constraints appears in Quinlan, (1983a). It should be noted that the constraints are weaker than the interpretations from which they are derived, and
hence the bounds computed via constraint propagation may be conservative. Constraint propagation is guaranteed to terminate, even with inconsistent information, by prohibiting the propagation of changes back to the initial source.

**An Example**

As an illustration of INFERNO's propagation method, consider the inference network in Figure 8. It is similar to the one used in the PROSPECTOR example and suggests attributes which may be relevant to whether a given candidate will be a good computer science student.

The network can be interpreted as follows. If a student has 4-unit mathematics then there is direct suggestive evidence that he will be a good computer science student. In INFERNO terminology, this becomes 4-U Math enables GoodCS with strength 0.7. If the student has both 4-unit maths and a high HSC score then he has analytical skills. That is, Analytic conjoins {HSC > 400, 4-U Math}. Note that there is no suggestion that these two attributes are independent, which conforms well with intuition. Finally, if a student is either analytical, or a home computer hobbyist, then he will be a good student. As expected, these two attributes are independent. In INFERNO, this would be expressed as the relation GoodCS disjoins-independent {Analytic, Hobbyist}.

Using this sample network, we can illustrate INFERNO's propagation method. If we are told that for a given student the probability that he has analytical skills is fairly high, say $P($Analytic$) > 0.6$, then $t($Analytic$) = 0.6$ and $f($Analytic$) = 0.3$. Using the conjoins constraint (3.1.3) and $t($Analytic$)$ we get $t($HSC > 400$) = 0.6$. By the disjoins-independent constraint (4.2.1), $t($GoodCS$) > 0.6$, as initially $t($Hobbyist$) = 0$. If we now find that this student has a good 4-unit maths result, i.e. P(4-U Math) is one, then $t($4-U Math$) = 1$ and hence $f($4-U Math$) = 0$. We can therefore derive $f($HSC > 400$) = 0.3$ (by (3.1.4), $f($Analytic$)$, $t($4-U Math$)$); $t($GoodCS$) = 0.7$ (by (1.1), $t($4-U Math$)$);

INFERNO would then produce the following summary:

- **'HSC > 400':** range 0.6-0.7
  - LB from $P('Analytic') > 0.6$, by conjoins
  - UB from $P('4-U Math') = true$, by conjoins

- **'4-U Math':** true by assumption

- **'Analytic':** range 0.6-0.7
  - LB by assumption
  - UB by assumption

- **'Hobbyist':** false from $P('GoodCS') > 0.7$, by disjoins-indep

- **'GoodCS':** range 0.7-1
  - LB from $P('4-U Math') = true$, by enables

**Handling Inconsistency**

INFERNO determines that information about a proposition is inconsistent if $t(A) + f(A) > 1$. To preserve consistency, INFERNO tries the two extreme possibilities, lowering $t(A) + f(A)$ to $1 - f(A)$, and $f(A)$ to $1 - t(A)$, and backtracks, trying to loosen antecedent bounds, either by weakening the strength of a relation, or by relaxing user-supplied bounds. One change is found for each proposition with inconsistent bounds, and a consolidated set of such changes is referred to as a 'rectification'. In general, there will be more than one possible change per proposition, so INFERNO considers the alternatives, and presents them to the user for consideration. Those that involve the least change ('reluctance') are presented first. The problem of finding rectifications is combinatorially explosive, and is controlled by a heuristic that discards partial rectifications of higher reluctance than the best rectifications so far.

**Summary**

The principal contribution of INFERNO is that it is the first system to formally recognise that the conditional independence assumption should not be made unless specifically advised that it is safe to do so. It is therefore able to guarantee that its (weak) conclusions are correct, in the light of the evidence. The ability to detect inconsistency in the evidence is invaluable.

INFERNO does not rely on an expert to estimate prior probabilities, although Quinlan acknowledges that INFERNO may not perform as well as a Bayesian system in the rare case where prior probabilities are calculable. However, this cannot be a serious criticism, as, if the statistical data is available to calculate priors, then there really cannot be a better method than the Bayesian one. Elsewhere Quinlan (1983b) notes that conflicts may arise when an attempt is made to map INFERNO's probability bounds to categorical results.

**DISCUSSION**

All of these plausible reasoning methods assume that it is reasonable to use some form of numerical
adjustment scheme to represent the validity of a proposition. In order to be computationally tractable, this usually necessitates the unjustifiable assumption of conditional independence among propositions (INFERNO excepted). This numerical representation has been questioned (Cohen and Grinberg, 1983), because the use of numbers tends to hide the reasoning process that produced them, and does not discriminate between different kinds of evidence, such as eyewitness versus hearsay. Inevitably there must be a loss of information whenever evidence is mapped onto numbers, no matter how mathematically valid is the procedure used to manipulate the numbers. Cohen and Grinberg propose the use of textual endorsements in place of numerical degrees of belief, and domain-dependent heuristics to propagate inferences, instead of combining functions. Unfortunately, there is as yet no practical system for using endorsements in this way. The appropriate choice of inference mechanism is largely determined by the nature of the problem domain.

The reader may be surprised at the relative simplicity of these plausible reasoning methods, and that knowledge-based systems can be constructed which utilise these techniques to achieve performance which is in some sense expert. However, it is now generally accepted that the real power of an expert system lies in its ability to utilise these techniques to achieve performance which is largely determined by the nature of the problem domain.

CONCLUSION
Several approaches to the problem of reasoning with uncertain knowledge have been reviewed, none of which claim to be universally applicable. Of these methods, the most promising would seem to be the Dempster-Shafer theory of evidence, as it includes PROSPECTOR'S Bayesian belief functions and MYCIN's certainty factors as special cases. It also is based on a more solid mathematical foundation than either PROSPECTOR or MYCIN. However, there is still a need for a method of handling evidence which is not independent, is highly contradictory, or which contains significant errors.

References


Plausible Reasoning


Biographical Note

John O'Neill holds a BE(Hons) from the University of New South Wales, and has been a Ph.D candidate in the Department of Computer Science at ADFA since August, 1984. Prior to joining ADFA, he was an experimental officer at the Royal Australian Naval Research Laboratory, and previously served as an engineer in the RAN. His research, which is supported by a cadetship from the Defence Science and Technology Organisation, is primarily in the area of knowledge acquisition for expert systems.

Helpers Needed

The editor is always looking for new names to add to the Journal's panel of referees and reviewers.

This is an 'equal opportunity' journal. We are particularly seeking volunteers from the non-academic majority, and also volunteers whose current interests are in one or more of the following: computer graphics, CAD/CAM, computer integrated manufacturing, image processing, expert systems, logic programming, information systems, simulation, in Pascal or C or Ada, and in the management of software projects.

If you are willing to contribute your services either as a book reviewer or as a referee for submitted technical papers, would you please write to the editor describing your interests, background and experience.

Would you also write if you wish to nominate a colleague for appointment to the Journal's Editorial Advisory Committee, or would like to comment on the existing membership of that body.

Currently the editor is:

Associate Professor J. Lions, Department of Computer Science, University of New South Wales, Kensington, NSW 2033
The Transition to Fifth Generation Technology: Conceptual Schema Implementation

G.M. McGrath†

An approach is detailed for transforming a binary-relationship conceptual schema into a semantically equivalent logic representation. The target schema is suitable for direct implementation with logic-based fifth generation technology. A powerful domain-independent inference mechanism and an information filter design for information base updates are presented. The information filter design emphasises run-time efficiency, while preserving the clarity of integrity constraint specifications in logic update procedures.

Keywords and Phrases: conceptual schema, logic (programming), fifth generation technology, deductive database.

CR Categories: H.2.1, I.2.3, I.2.4

1. INTRODUCTION

Logic, implemented as Prolog, is a very high-level programming language which is being used increasingly as a general-purpose systems development tool (Dahl, 1982). Logic is also the kernel language of the Japanese Fifth Generation Computer Systems project where it will be implemented at the machine level, within highly parallel inference machines (Furukawa and Yokoi, 1984). NIAM (Nijssen's Information Analysis Method) (Verheijen and Van Bekkum, 1982) is an information systems design methodology and information modelling tool which has evolved from the work of Falkenberg (1976) and Nijssen (1977). It is based on binary relationships, has its origins in formal logic (Jardine, 1984) and, in this paper, we demonstrate that a NIAM conceptual schema can, in a convenient and natural way, be transformed to a logic representation suitable for implementation with fifth generation technology. Thus the logic schema may be viewed as both an internal schema and an alternate form of representation at the conceptual level. This is a direct consequence of the dual interpretations of logic programs as problem specifications and working programs (Kowalski, 1979a).

The approach we have taken was largely determined by the following criteria:

1. The (target) logic schema must be implementable on logic-based technology, thus greatly simplifying the implementation of applications built around corporate data models and preparing the way for a smooth transition to the fifth generation technology planned for the early 1990's. More immediately, in restricted cases, applications can be implemented using present-day Prolog software (McGrath and Debenham, 1984).

2. The target schema must provide a clear representation of the NIAM original, yet be in a form which can be manipulated conveniently and processed efficiently.

3. The target schema must represent all information relevant to a 'Universe of Discourse' (or application) in the one place and form. Hence, in common with Flynn and Laender (1985), our target schemas might be used to determine redundancies and inconsistencies in existing data base systems.

4. The target schema must be 'minimal' in its application specific component; common inference and validation rules will be included in all schemas.

The general area of graphical front-ends to deductive databases has received considerable attention of late. The reader is referred particularly to the work of Dart and Zobel (1985), in which a mapping from a new conceptual schema, LOCS, to logic is described. LOCS (Logic Oriented Conceptual Schema) is based on NIAM and has proved useful as a conceptual schema design tool for both logic programmers and for general users unfamiliar with logic programming principles. LOCS, however, includes only a limited
subset of NIAM primitives and, in its present form, is less suitable than our approach as a vehicle for converting existing systems to fifth generation technology. This is an underlying motive for our work and the implementation detailed here is consistent with the new generation data base architecture described by Griethuysen (1982).

Our Prolog notation and some background are given in Section 2 and NIAM features are introduced in Section 3. Details of the NIAM to logic transformation are presented in Section 4. Data base interactions in our implementation are discussed in Section 5 and Section 6 contains concluding comments.

2. NOTATION AND BACKGROUND

While not essential, a limited understanding of logic programming concepts (Kowalski, 1979b) and of conceptual schema construction principles (Griethuysen, 1982) will assist the reader. In addition, some appreciation of the relationship between logic programming and fifth generation computer systems (Shapiro, 1983) and of the ‘Intelligent Knowledge Base System’ architecture proposed by Nijssen (1983) would help to place this work in perspective. Our notation differs slightly from the widely-used Edinburgh Prolog (Clocksin and Mellish, 1981). A clause

\[ P : \neg Q, R, S \]

where \( P, Q, R \) and \( S \) are predicates, may be interpreted declaratively as ‘\( P \) is true if \( Q \) and \( R \) and \( S \) are all true’. Alternatively, \( P \) may be interpreted as a procedure with \( Q, R \) and \( S \) as its calls. A predicate, \( p(t_1, \ldots, t_n) \), consists of a (lower case) predicate symbol \( p \) and the sequence of terms \( t_1, \ldots, t_n \). Terms may be variables (signified by the lower case letters \( x, y, z \) and \( w \) with (optional) numeric suffixes), constants (enclosed in quotes where there is any possibility of ambiguity) or compound. Compound terms consist of a (lower case) functor and a sequence of terms enclosed in brackets, each of which (again) may be variable, constant or compound. A special case of a compound term is \([X_1|X]\) which denotes a list with first element \( X_1 \) and remainder \( X \). We use the underline symbol ‘_’ to denote ‘don’t care’ variables and \( p_x \) where we wish to treat a functor as a variable (as allowed in a number of Prolog implementations).

3. NIAM CONCEPTUAL SCHEMATA

The three basic components of a NIAM conceptual schema (Nijssen, 1983) are specifications of: information base sentence types, constraints on information base states and transitions, and inferences on the information base.

We now provide a brief overview of NIAM sentence type definition and constraint constructs. Inference is discussed in detail in Section 5.2.

A NIAM information base consists of a set of deep structure elementary sentences. A conceptual schema sentence type is represented graphically in an Information Structure Diagram (ISD) as illustrated in Figure 1.

\[
\text{Figure 1. NIAM Sentence Type}
\]

\( N_1 \) and \( N_2 \) are object or entity types, referred to as non-lexical object types (NOLOTs), while \( L_1 \) and \( L_2 \) are label types, referred to as lexical object types (LOTs). Individual occurrences of NOLOTs are called lexical object instances (LOIs). \( p \) and \( p' \) are predicate names (sometimes called roles) where \( p \) is referred to as the converse of \( p \). The association \( <N_1, p, N_2> \) is called an idea type and the brackets around \( L_1 \) and \( L_2 \) indicate implicit associations \( <N_1, with, L_1> \) and \( <N_2, with, L_2> \) called bridge types. We will sometimes identify sentence types by their idea types and we call a sentence that includes a converse predicate symbol a converse sentence.

For example, Figure 2 contains a sentence type which includes the idea type \(<\text{Location, is-in, City}>\) and the bridge types \(<\text{Location, with, Lname}>\) and \(<\text{City, with, Cname}>\). If Madison Square Garden and New York are instances of Location and City respectively, then the information base will contain the sentence:

The Location with Lname Madison Square Garden is-in the City with Cname New York.

Since the converse of ‘is-in’ is ‘contains’, the information base will also include the (converse) sentence:

The City with Cname New York contains the Location with Lname Madison Square Garden.

The above represent the simplest kind of sentence type. In Section 4.1 we introduce a variation that, while slightly more complex, is capable of representing a much wider range of NIAM sentences.

Verheijen and Van Bekkum (1982) identify six NIAM ISD constraint types:

<table>
<thead>
<tr>
<th>symbol</th>
<th>type of constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \leftarrow )</td>
<td>identifier</td>
</tr>
<tr>
<td>( \subseteq )</td>
<td>uniqueness</td>
</tr>
<tr>
<td>( \subseteq )</td>
<td>subset</td>
</tr>
<tr>
<td></td>
<td>equality</td>
</tr>
<tr>
<td>( \supseteq )</td>
<td>disjoint</td>
</tr>
<tr>
<td>( \leftarrow )</td>
<td>total role</td>
</tr>
</tbody>
</table>

An example of each of these constraint types is included in Figure 2, which is part of a conceptual
schema for an information system for recording and analysing results of international track and field meetings.

An identifier constraint constrains the degree of association between NOLOTs to be 1:1 (a meet is held at only one location), 1:n (one location can host many meets) or m:n (an athlete can compete in many events and the one event can be contested by many athletes). A uniqueness constraint specifies that an occurrence of a NOLOT can be uniquely identified by a combination of role occurrences (marks can be identified by associations with Athlete, Event and Meet NOLOTs). A subset constraint specifies that the population of one role must be the subset of the population of another role (event record holders must compete in that event). An equality constraint specifies that if one role is played, then so must one or more other roles (a location is in a city and also in a nation). A disjoint constraint specifies that the populations of two subsystems exclude each other (a mark can be made in a track event or a field event but not both). A total role constraint demands that all occurrences of a NOLOT play a particular role (all marks must be made in meets).

One further important NIAM construct is the subtype, indicated by an arrow directed from subtype to supertype (Track-mark and Field-mark are subtypes of the supertype Mark). A subtype inherits all properties of its supertype and may have additional properties of its own. Subtypes of the one supertype need not be disjoint.

4. NIAM: LOGIC REPRESENTATION

4.1 Sentence Type Definition
Lafferty (1985) presents rules which allow any NIAM schema to be transformed into a set of subschemas with the structure given in Figure 3. The details of the transformation process are presented elsewhere (McGrath, 1986). Here we just note that transformation typically involves the creation of additional composite NOLOTs \( N_c \) in Figure 3 and that our structures can be mapped into optimum normal form relations (Nijssen, 1985). Hence, we shall refer to our target structures as normalized. An example of a normalized structure, derived from the schema in Figure 2, is presented in Figure 4.

The normalized structure in Figure 3 contains \( 2n \) sentence types, defined in our logic conceptual schema as the assertions
The Transition to Fifth Generation Technology

4.2 Constraints

Occurrences of identifier, subset, equality, disjoint and total role constraints are specified in the logic conceptual schema using the predicate names mapping, subsumed-by, coexistent, exclusive and mandatory respectively. Examples are

- `mapping([[Location, is-in, City]], 1:1)`
- `subsumed-by([[Athlete, holds-record-for, Event]], [Athlete, competes-in, Event])`
- `coexistent([[Location, is-in, City]], [Location, is-in, Nation])`
- `exclusive([[Track-mark, Field-Mark], subtypes])`
- `mandatory([[Meet, held-at, Location], role])`

No application specific specifications of uniqueness constraints are necessary and, as Flynn and Laender...
The Transition to Fifth Generation Technology

(1985) have noted, uniqueness constraints are, in a sense, not constraints at all, but simply a means of identifying objects which require compound identifiers (this information is implicit in our sentence type definitions). However, application independent rules apply to all other constraint types. These rules are also the validation procedures which prevent information base corruption during updates and can be activated by the goal `? :- invalid(x, y)` where corrupt data will be returned as x and y denotes the reason for the corruption. For example

\[
\text{invalid}(x, \text{identifier}) :- \text{isa}(x, \text{sentence}), \\
\text{another-instance}(y, x)
\]

\[
\text{invalid}([\text{id-by}(x, z), p_y, y], \text{identifier}) :- \\
\text{mapping}([x, p_x, y], 1:1), \\
\text{isa}(p_x(\text{id-by}(x, z)), \text{with}(y, z)), \text{sentence),} \\
\text{isa}(p_y(\text{id-by}(x, z)), \text{with}(y, z_2)), \text{sentence,} \\
\text{different}(z_1, z_2).
\]

That is, we do not allow the absence of an identifier constraint (more than one instance of a sentence) or \(m:n\) mappings between entity types. Hence, we only declare 1:1 mappings in the conceptual schema, with any unspecified mappings (i.e. all converses) being 1:n.

Similar rules apply to subset, equality, disjoint and total role constraints. These rules include (as conditions) the application specific assertions (subsumed-by etc.) referred to above.

In a logic data base, validation rules are generally most naturally expressed as static constraints and this is the form we have used above. While the use of transition constraints is sometimes unavoidable (see Section 5.4), their declarative reading can be opaque. Transition constraints, though, can be used to prevent data base corruption at update time, whereas static constraints are generally of more use in detecting errors after the event. In Section 5.3 we detail an update strategy that employs static constraints but activates validity checks prior to updates.

5. INFORMATION BASE INTERACTIONS

5.1 System Architecture

Our architecture, which is based on the ISO view of a general architecture for information systems (Griethuysen, 1982), is illustrated in Figure 6. While we have implemented a prototype (in micro-Prolog (McCabe and Clark, 1980)), its usefulness is limited by all system components being simulated in main memory. For most non-trivial applications a Prolog-DBMS interface of the type described by Jarke, Clifford and Vassiliou (1984) is deemed essential.

The CIP (Conceptual Information Processor) controls all interactions with the information system and, in Nijssen’s extension of the ISO architecture (Nijssen, 1983) it is broken down into two major components; an information filter and an information amplifier. The principal responsibility of the information filter is to protect the integrity of the information base (i.e. it ensures that only update transactions consistent with schema definitions are accepted) while the information amplifier is responsible for retrieving and inferring information from the information base. We provide further detail on these two components in the remainder of this section.

5.2 Retrievals

All sentences explicitly represented in the information base can be retrieved by activating the goal

\`
\text{? :- isa}(x, \text{sentence}).
\`

In addition, all converse sentences will be retrieved as part of this operation (using the ‘converse’ rule presented in Section 4.1). This is a simple case of inference and below we demonstrate that an information amplifier, composed of a logic programming inference mechanism supplemented with a few simple conceptual schema rules, can be used to derive all binary relationships between instances of different information base entity types.

First, we specify another form of standard information base sentence structure. Since, in normalized structures, non-composite NOLOTs are uniquely identified by one LOT, we may derive an alternate (virtual) sentence representation using the rules

\[
\text{isa}(p_x(_, \text{id-by}(x, [\text{with}(x, \text{of}(y, z)]))), \text{sentence}) :- \\
\text{isa}(p_x(_, \text{with}(x, \text{of}(y, z))), \text{sentence})
\]

and

\[
\text{isa}(p_x(\text{id-by}(x, [\text{with}(x, \text{of}(y, z)])), _), \text{sentence}) :- \\
\text{isa}(p_x(\text{with}(x, \text{of}(y, z)), _), \text{sentence})
\]

the second rule being for the converse case.
Now consider the subschema in Figure 7. We could derive relationships between Athlete and Meet entities using the rule

\[
\text{related(id-by(Athlete, x), id-by(Meet, y)) :-}
\]

\[
\text{isa(made-by(id-by(Mark, z), id-by(Athlete, x)), sentence),}
\]

\[
\text{isa(made-at(id-by(Mark, z), id-by(Meet, y)), sentence)}
\]

whereas we can derive relationships between Athlete and Location entities using

\[
\text{related(id-by(Athlete, x), id-by(Location, y)) :-}
\]

\[
\text{isa(made-by(id-by(Mark, z), id-by(Athlete, x)), sentence),}
\]

\[
\text{isa(made-at(id-by(Mark, z), id-by(Meet, w)), sentence),}
\]

\[
\text{isa(held-at(id-by(Meet, w), id-by(Location, y)), sentence).}
\]

However, by using the rules

\[
\text{related(id-by(x_1, x_2), id-by(y_1, y_2)) :- (R1)}
\]

\[
\text{isa(p_x(id-(by(x_1, x_2), id-by(y_1, y_2)), sentence) and}
\]

\[
\text{related(x, y) :- isa(p_x(x, z), sentence), (R2)}
\]

\[
\text{related(z, y)}
\]

we can retrieve all virtual binary relations (Athlete-Meet, Athlete-Location and Location-Meet) as well as converses and relationships between entities in sentences explicitly represented in the information base. Furthermore, this applies to any normalized structure. An informal proof follows.

All entity types (nodes) in a normalized conceptual schema are connected and it is possible to traverse any normalized structure from any given node to all other nodes in the network. Rule (R2) above ensures that, if a relationship can be retrieved between an entity \(x\) and another entity \(z\), \(n\) nodes distant, a relationship can also be retrieved between \(x\) and entity \(y\), connected to \(z\) and \(n + 1\) nodes distant from \(x\). However, by rule (R1), all relationships between instances of entity types one node apart can be retrieved. Hence, by induction, all binary relationships (between instances of different entity types) can be retrieved for any normalized structure.

\(n\)-ary relationships can be 'collected' (as lists) as a schema is traversed using the rules

\[
\text{related([x | y]) :- related(x, y)}
\]

(R3)

and

\[
\text{related([x_1, x_2 | y]) :- related(x_1, x_2), related([x_2 | y]).}
\]

(R4)

Thus, we have detailed a powerful, yet extremely simple, inference mechanism. The information amplifier is available as 'off the shelf' software and four application independent rules (which double as implementable code) make up the conceptual schema inference rules component. The only other code required is the procedures for deriving converses and the alternate sentence form presented above (included in the sentence type definition component of the conceptual schema).

Note though, that with most current Prolog implementations (including ours), our inference procedures will loop in some circumstances. (If a route exists between nodes \(x\) and \(y\) then our converse rules ensure there is also a route from \(y\) to \(x\). By using (R2) repeatedly, the inference mechanism can be locked into a path \([x, \ldots, y, \ldots, x, \ldots, y, \ldots]\). Loops can be avoided by reformulating the inference rules to include a loop detection mechanism (Kowalski, 1983). In our reformulation, we use a technique commonly used with current Prolog implementations (see e.g. Stirling's (1984) solution to the plan-formation problem). This involves using an extra argument to collect (as a list) all nodes passed through up to any point during query resolution, so as to avoid looping through nodes previously traversed.

Finally, while we are not concerned with external presentation aspects here, if derived relationships are to be presented in deep structure sentence form, the conceptual schema must contain a pre-defined (and application dependent) set of declarations linking predicate names with (virtual) relationships between entity types.

5.3 Updates

A transaction can be viewed as a series of operations causing an information base to pass through a sequence of states, \(IB_1, \ldots, IB_n\), with the requirement that only the initial and final states must be valid (Furtado, 1983). At the conceptual level, information base sentences are indivisible and, hence, cannot be modified; only added to or deleted from an information base (note that this does not preclude an external view which allows modifications). Thus a single-phase transaction causes an information base to pass through one state change only and can be either an 'add sentence' or 'delete sentence' operation, whereas a multi-phase transaction results in more than one state change and consists of a set of single-phase transactions.

Previously, we noted that static constraints are of more use in detecting than preventing integrity violations. In addition, they may require unacceptable amounts of processing resources and time when applied. For example, naive invocation of the 'invalid' procedures discussed in Section 4.2 would, quite probably, be disastrous in a non-trivial on-line application. At the same time, alternatives to static constraints (e.g. constraint specifications that refer explicitly to transaction and information base state contents) are less clear in their declarative semantics.

The solution we propose involves taking a relevant snapshot of the conceptual schema and information base. When an update transaction is processed, the
SELECTOR module in the information filter (see Figure 6) activates selection procedures which then infer which information base sentences and conceptual schema constraints are relevant to the update. Relevant sentences and constraints are then placed in a temporary buffer and the ENFORCER module initiates a validity check through the goal

\[ ?: - invalid(x, y)' \].

If the goal succeeds, the update fails and the user is advised of the reason. Otherwise, the transaction operations are applied to the information base. A CONTROL module in the information filter is responsible for coordinating the update, including transaction receipt, activation of selection and validation procedures, actual information base update and despatch of the result to the user.

Our constraint selection approach is essentially an adaption (to what is, in effect, a binary-relational data base with limited constraint types) of a method proposed by Nicolas (1982) for using logic for integrity constraint checking in relational data bases. We now describe our approach.

Assume that we wish to add the sentence

\[ p(\text{id-by}(A_c, a), \text{id-by}(B, b)) \]

to the information base (note that we use the alternate form of sentence representation introduced in Section 5.2). We call this the update sentence. A constraint can only be violated if it applies to the subschema given in Figure 8. Thus, the SELECTOR module will only extract a conceptual schema constraint declaration (see Section 4.2) for the relevant snapshot if it contains a term \([A_c, p, B], [A_c, q_i, C_i] \text{ or } [C_i, q_i, A_c] \text{ for } i = 1, \ldots, n\), and \([B, r_j, D_j]\) or \([D_j, r_j, B]\) for \(j = 1, \ldots, m\). Disjoint constraint specifications will only be selected if they include the NOLOTs, \(A_c\) or \(B\). In addition, SELECTOR will retrieve from the information base the four sets:

\[ \neg \{p(\text{id-by}(A_c, x), \text{id-by}(B, y)) \mid x = a\} \]
\[ \neg \{q_i(\text{id-by}(A_c, x), \text{id-by}(C_i, y)) \mid x = a\} \text{ for each } i = 1, \ldots, n \]
\[ \neg \{r_j(\text{id-by}(B, x), \text{id-by}(D_j, y)) \mid x = b\} \text{ for each } j = 1, \ldots, m \]
\[ \neg \{s(\text{id-by}(E, x), y) \mid (x = a \text{ and } A_c \text{ and } E \text{ are disjoint subtypes}) \text{ or } (x = b \text{ and } E \text{ and } B \text{ are disjoint subtypes})\} \]

The update sentence completes the relevant snapshot for the add operation. The first step in a delete operation is to establish that the update sentence is present in the information base. If not, the update fails immediately. If it does exist, the relevant snapshot is identical to that extracted for an add, minus the update sentence.

In general, our approach substantially reduces the cost of constraint checking during information base updates. For example, in adding a sentence of the type \(<\text{Meet}, \text{held-at}, \text{Location}>\) to an information base that is a model of the conceptual schema in Figure 2, only 10 out of 32 constraints and five out of 14 sentence types will be included in the relevant snapshot.

Note though that our selection criteria can be refined further. For example, referring again to Figure 8 and the addition of a sentence of type \(<A_c, p, B>\), sentences of type \(<A_c, q_i, D_j\) are not relevant if the role \(<A_c, p, B>\) subsumes the role \(<A_c, q_i, D_j\). If the reverse constraint applies though (i.e. \(<A_c, q_i, D_j\) subsumes \(<A_c, p, B>\)), these sentences are relevant. Furthermore, at present, our implementation excludes application of constraints to derived information (i.e. to virtual data outside the defined sentence types of the information base). Constraint checking and enforcement in deductive databases is a subject that has generated considerable interest of late (see e.g. Lloyd and Topor, 1985; Asirelli, Santis and Martelli, 1985). We note that Topor, Keddis and Wright (1985) have implemented an extension of Nicolas’ method in a deductive data base system.

Refinements of the type outlined above and integrity checks on derived information could both be usefully included in our implementation. At this stage, however, our first priority is extending our system to include more recently proposed NIAM constraint types (see e.g. Lafferty, 1985) beyond those contained in the early proposal of Verheijen and Van Bekkum (1982).

### 5.4 Transition Constraints

The ISD is intended primarily as a graphical tool to assist in conceptual schema design and is not capable of expressing all features of an information system. In particular, transition constraints cannot generally be represented in an ISD.

---

**Figure 8. Update Subschema**
The Transition to Fifth Generation Technology

Consider the schema in Figure 9. Outstanding faults are assigned to technicians who are 'idle' (Wcode = I), at which stage their status changes to 'working' (Wcode = W). A technician is assigned only one fault at a time and upon fixing a fault the technician's status must return again to 'idle'. This is an example of a transition constraint and it could be modelled outside the ISD using, for example, Petri-nets or Activity Graphs. In logic, we can express the constraint as

\[
\text{invalid([Technician, has, Status], transition)} : - \text{changes-to(assigned-to}(x, y), \text{fixed-by}(x, y)), \text{not changes-to(has}(y, \text{with}(\text{Status, of}(Wcode, W))), \text{has}(y, \text{with}(\text{Status, of}(Wcode, I))))
\]

and

\[
\text{changes-to}(x, y) : - \text{is-in}(x, IB_{old}), \text{not is-in}(x, IB_{new}), \text{not is-in}(y, IB_{old}), \text{is-in}(y, IB_{new})
\]

where IB_{old} and IB_{new} denote the old and new information base states respectively.

Application of this validity check requires an extension to the CIP architecture outlined previously, in that we have to provide temporary buffers to hold snapshots of both the old and new information bases. The significant aspect though, is that information modelled in NIAM as an ISD complemented with, for example, a Petri-net, is now specified in one form only. This is consistent with our previously stated aim of providing a target representation which facilitates schemas comparisons.

6. CONCLUSION

We have demonstrated that NIAM conceptual schemas may be mapped to target logic specifications suitable for direct implementation on fifth generation technology. In addition, we have outlined a conceptual information processor design that allows efficient information base updates, while not sacrificing the clarity of integrity constraints, and takes natural advantage of the underlying technology in its inference component. Our logic schemas are in a form suitable for implementation on logic-based fifth generation technology or as fifth generation prototypes using present-day Prolog software.

ACKNOWLEDGEMENTS

The permission of the Chief General Manager, Telecom Australia, to publish this paper is acknowledged. I would like to thank John Debenham for comments on an earlier related paper that helped determine the direction of this work.

References


The Transition to Fifth Generation Technology


Biographical Note

Mike McGrath is a Senior Project Manager in the Systems Development Section of Telecom's Information Systems Branch in New South Wales. He obtained his Science degree from the University of Melbourne in 1969 and his Master's degree in Computing for the New South Wales Institute of Technology in 1982. His current research interests include data base integration and systems conversion problems.

ACC87 — MBA
Australian Computer Conference, 1987
Melbourne, Victoria, 8-11 September, 1987

CALL FOR PAPERS

The annual conference of the Australian Computer Society will be held in Melbourne in September.

Papers reporting on interesting and exciting aspects of computing in the areas of Manufacturing, Banking and Artificial Intelligence are now solicited.

The Australian Computer Conference is held once each year and attracts something like 5-10% of the Society's 12,000 members. Why they come is fairly clear. The conferences are designed to update members with the latest and best in research, innovation and achievement.

Draft versions are needed by 28 March so that notifications for acceptance can be made by 30 April. Final copies of accepted papers are due by 30 June 1987.

All papers will be refereed. For further information please contact:

Pearl Levin,
Program Chairman, ACC87-MBA,
Chisholm Institute of Technology,
PO Box 197, Caulfield East, Victoria, Australia
(03) 573-2184

Integrating the Structured Analysis and Design Models: A Category-Theoretic Approach

T. H. Tse†

A number of models have been proposed under the name of structured analysis and design. It has been pointed out, however, that there is no common theoretical framework among them. Transformation of a specification from one model to another, although often recommended by authors, can only be done manually. A category-theoretic approach is proposed in this paper. As a result, development of structured specifications can be assisted through structured tasks and morphisms, and the integration of structured models can be achieved through functors and free categories.

Keywords and Phrases: Keywords and phrases: category theory, structured analysis, structured design
CR Categories: D. 2.1, D. 3.1, G. 2, K. 6.3

1. INTRODUCTION
A number of models have been proposed under the name of structured analysis and design. Examples are data flow diagrams (DeMarco, 1978; Gane and Sarson, 1979; Weinberg, 1980), Jackson structure diagrams, Jackson structure text (Jackson, 1975), system specification diagrams, system implementation diagrams (Jackson, 1983), Warnier/Orr diagrams (Orr, 1977) and structure charts (Yourdon and Constantine, 1979). They are widely accepted by practicing systems analysts and designers through the simplicity of use and the ease of communication with users.

It has been pointed out by Tse (1986), however, that there is no common theoretical framework among them. Transformation of a specification from one model to another, although often recommended by authors, can only be done manually. An initial algebra approach has been proposed in that paper to integrate these models. Given a specification in one structured model, the initial algebra approach provides a formal means of mapping it to an equivalent specification in terms of another model. It does not, however, provide a means of developing the specification in the first place.

A category-theoretic alternative is proposed in this paper. The concept of an elementary task is defined. Intuitively, it contains a process together with its input and output states. Structured tasks are defined as elementary tasks linked up by five operations: sequence, selection, parallelism, iteration and procedure call. The usual structured models such as DeMarco data flow diagrams, Yourdon structure charts or Jackson structure text can be seen as different ways of representing structured tasks. Morphisms, or functions preserving the structures, can be defined between tasks. The usual refinements in structured methodologies can be seen as the reverse of morphisms. These concepts help a system developer visualize the internal structure of a system, assemble or refine subsystems, and verify the consistency and completeness of a design.

Furthermore, we can integrate the models by providing mappings from one type of specification to another via the structured tasks. A DeMarco or Yourdon specification, for example, can be mapped to a structured task specification by means of a functor. Conversely, given a structured task specification, free categories of DeMarco or Yourdon specifications will result.

2. ADVANTAGES OF A CATEGORY-THEORETIC APPROACH
There are several advantages of using the category-theoretic approach to integrate the structured analysis and design models:

a. Many attempts have already been made to computerize the system development environment. Examples are ISDOS (Teichroew and Hershey,
3. A BRIEF INTRODUCTION TO CATEGORY THEORY

In this section, we will give a brief introduction to the fundamental concepts in category theory. Interested readers may refer to Arbib and Manes (1975a; 1975b), Goguen et al. (1973; 1975; 1976), Goldblatt (1984) and Mac Lane (1971) for further details.

The concepts of categories and functors are stronger than the concepts of sets and functions in set theory. A category consists of a collection of objects, together with relationships amongst these objects called morphisms. Functions can be defined mapping objects in one category to those of another category. In particular, the most useful of these functions also preserve the morphisms amongst objects, and they are known as functors. More formally, a category \( X \) consists of

a. a class of objects in \( X \);
b. for each pair of objects \( A \) and \( B \) in \( X \), a set of morphisms \( f: A \rightarrow B \) subject to the following conditions:

1. For each pair of morphisms \( f: A \rightarrow B \) and \( g: B \rightarrow C \), there exists a morphism \( g \circ f: A \rightarrow C \), which is called the composite of \( f \) and \( g \).
2. Given any morphisms \( f: A \rightarrow B \), \( g: B \rightarrow C \) and \( h: C \rightarrow D \), \( h \circ (g \circ f) = (h \circ g) \circ f \).
3. For each object \( A \) of \( X \), there exists a morphism \( i_A: A \rightarrow A \), called an identity morphism.
4. Given any morphism \( f: A \rightarrow B \), \( f \circ i_A = i_B \circ f \).

Mappings between categories preserving the morphisms are called functors. More formally, for any functor \( F: X \rightarrow Y \),

a. each object \( A \) in \( X \) is related to one and only one object \( A \) in \( Y \), denoted by \( A = F(A) \);
b. each morphism \( f: A \rightarrow B \) in \( X \) is related to one and only one morphism \( F(f): F(A) \rightarrow F(B) \) in \( Y \)

such that compositions and identities are preserved. That is to say,

1. if \( h = g \circ f \), then \( F(h) = F(g) \circ F(f) \);
2. \( F(i_A) = i_{F(A)} \).

4. STRUCTURED TASKS

Let us apply the categorical concepts to structured models. We define a flow as a tuple

\(<s_{11}, s_{12}, \ldots, p, s_{21}, s_{22}, \ldots>\>

where \( p \) is a process, \( s_{11}, s_{12}, \ldots \) are input states and \( s_{21}, s_{22}, \ldots \) are output states. Any standard process \( p \) should have only one input state \( s_1 \) and one output state \( s_2 \). The flow corresponding to a standard process will therefore be of the form \(<s_1, p, s_2>\).

Besides the standard processes, we will also define auxiliary processes which may have more than one input or output states. There are three auxiliary processes:

a. A decision, which has one input state \( s_1 \) and two output states \( s_2 \) and \( s_3 \). The flow corresponding to a decision is denoted by

b. A collection, which has two input states \( s_1 \) and \( s_2 \) and one output state \( s_3 \). The flow corresponding to a collection is denoted by 
\(<s_1, \oplus, s_2, s_3>\).

c. A fork, which also has one input state \( s_1 \) and two output states \( s_2 \) and \( s_3 \). The flow corresponding to a fork is denoted by 
\(<s_1, \otimes, s_2, s_3>\).

d. A join, which also has two input states \( s_1 \) and \( s_2 \) and one output state \( s_3 \). The flow corresponding to a join is denoted by 
\(<s_1, \otimes, s_2, s_3>\).

We define an elementary task as a set containing one and only one flow. A task in general is defined as either an empty set or a set containing one or more flows.

In structured models, we are more interested in tasks which are in the form of sequences, selections, iterations, parallelisms or procedure calls of other tasks. Such tasks are called structured tasks, and are defined recursively as follows:

a. Any elementary task \( T = \{<s_1, p, s_2>\} \) is structured. We will denote this structured task by \( T(s_1, s_2) \).

b. The sequence of any two structured tasks \( T_1(s_1, s_2) \) and \( T_2(s_2, s_3) \), defined as
\[(T_1 \cdot T_2)(s_1, s_3) = T_1(s_1, s_2) \cup T_2(s_2, s_3),\]
is a structured task.

c. The selection of any two structured tasks \( T_1(s_1, s_2) \) and \( T_2(s_3, s_4) \), defined as
\[T_1(s_1, s_2) \cup T_2(s_3, s_4) \cup \{<s_2, s_4>, \otimes, s_6>\},\]
is a structured task.

d. The parallelism of any two structured tasks \( T_1(s_1, s_2) \) and \( T_2(s_3, s_4) \), defined as
\[T_1(s_1, s_2) \cup T_2(s_3, s_4) \cup \{<s_2, s_4>, \otimes, s_6>\},\]
is a structured task.

e. The iteration of any structured task \( T(s_1, s_2) \), defined as
\[(\times(T))(s_3, s_4) = \{<s_3, s_4>, \oplus, s_4>\} \cup T(s_1, s_2) \cup \{<s_3, s_4>, \oplus, s_4>\},\]
is a structured task.

f. The procedure call of any structured task \( T(s_1, s_2) \), defined as
\[(\times(T))(s_3, s_4) = \{<s_3, \otimes, s_1, s_4>\} \cup T(s_1, s_2) \cup \{<s_3, s_4>, \otimes, s_4>\},\]
is a structured task.

Figure 1. Diagrammatic Representation of Structured Tasks

To ease user understanding, we will represent structured tasks by task diagrams, as shown in Figure 1. Task diagrams are, in fact, a variation of DeMarco data flow diagrams. We simply forget about the input/output types and insist that every selection or parallelism must have a single input state and a single output state, as shown in the example of Figure 2.

5. STRUCTURED FUNCTIONS AND MORPHISMS

Given two tasks (or sets of flows) \( T_1 \) and \( T_2 \), various functions \( f: T_1 \rightarrow T_2 \) can be defined mapping individual flows of one task to those of the other. An important class of these functions is known as structured functions, which preserve the structuredness of the subtasks in \( T_1 \) and \( T_2 \). Formally, a structured function \( f \) must satisfy the following condition for any subtask \( T \) in \( T_1 \):

\( T \) is structured if and only if \( f[T] \) is structured.
We are interested in actually constructing a structured function. This can be done through abstractions and morphisms. A function \( f : T_1 \to T_2 \) is an abstraction if and only if there exist a structured subtask \( T \) in \( T_1 \) and an elementary subtask \( \{ <s_1, p, s_2> \} \) in \( T_2 \) such that:

a. any flow in \( T \) is mapped to the single flow \( <s_1, p, s_2> \) in \( T_2 \);

b. any other flow in \( T_1 \) is mapped to the same flow in \( T_2 \).

Morphisms are then defined between tasks using the following recursive definition:

a. An abstraction is a morphism.

b. Compositions of morphisms are morphisms.

It can be shown that any morphism must be a structured function. In other words, a morphism preserves the structures of the tasks.

We will relate the concept of morphisms to the usual concept of refinements in structured analysis and design. A task \( T_1 \) is said to be a refinement of another task \( T_2 \) if and only if there exists a morphism \( f \) mapping \( T_1 \) on to \( T_2 \). In this way, morphisms and refinements help the users to visualize the development of a specification. The sample specification in Tse (1986), for instance, can be developed in terms of morphisms and refinements. Thus, using the obvious morphisms,

\[
\text{process-sales} \uparrow \text{MORPHISM} \downarrow \text{REFINEMENT} \downarrow \text{process-sales} \seq \text{get-valid-order; get-order; put-invoice; process-order; process-sales end}
\]

\[
\text{process-sales} \uparrow \text{MORPHISM} \downarrow \text{REFINEMENT} \downarrow \text{process-sales} \seq \text{get-valid-order seq get-order; validate-order; get-valid-order end; process-order sel process-local-order seq prepare-local-inverse; compute-tax; process-local-order end; process-order alt prepare-overseas-inverse process-order end; put-issue process-sales end}
\]
Design Models: A Category-Theoretic Approach

Figure 5. Morphisms and Refinements in Yourdon Structure Charts

\[ T_1 = \{ \text{<customer-info, process-sales, invoice>} \} \]

can be refined to

\[ T_2 = \{ \text{<customer-info, get-valid-order, valid-order>}, \]
\[ \text{<valid-order, process-order, invoice-info>}, \]
\[ \text{<invoice-info, put-invoice, invoice>} \} \]

which can be further refined to

\[ T_3 = \{ \text{<customer-info, get-order, order>}, \]
\[ \text{<order, validate-order, valid-order>}, \]
\[ \text{<valid-order, order, os-order>}, \]
\[ \text{<lc-order, prepare-local-invoice, pre-tax-info>}, \]
\[ \text{<pre-tax-info, compute-tax, lc-invoice-info>} \],
\[ \text{<os-order, prepare-overseas-invoice, os-invoice-info>}, \]
\[ \text{<lc-invoice-info, ls-invoice-info>, invoice-info>,} \]
\[ \text{<invoice-info, put-invoice, invoice>} \} \]

and so on. In this way, stepwise refinement can simply be conceived as a set-manipulation process, which can easily be aided by a computerized system. For the sake of user friendliness, however, we can also represent the refinement steps diagrammatically, as shown in Figure 3.

6. FUNCTORS AND FREENESS
If we modify our structured tasks and morphisms slightly, we will obtain the common structured models. For example, if each of the input and output states is given a type such as source, sink, file or data, then the structured tasks can be represented diagrammatically by DeMarco data flow diagrams. If we forget about the names of input and output states, we can represent the structured tasks by Jackson structure text, as shown in Figure 4. If all the abstractions are documented, one level at a time, we can represent the structured tasks by Yourdon structure chart, as shown in Figure 5.
7. CONCLUSION

Since the structured models are slightly different ways of representing structured tasks, we would like to know whether there is any mapping which helps us convert one representation to another. It can be shown that the structured tasks and the morphisms form a category. Functors, or functions which preserve the morphisms, can be defined between categories. A Yourdon specification, for example, can be mapped to a task diagram specification by means of a functor. In other words, given a Yourdon specification, one and only one structured task specification will result. Conversely, a structured task specification can be mapped to free categories of Yourdon specifications. In this case, the system developer will be presented with possible design choices and may select the appropriate implementation based on personal experience, environmental considerations and hardware characteristics. Functors and free categories are illustrated in Figure 6.

The arguments for DeMarco and Jackson specifications are similar. As a result, the structured models can be linked up through the paths shown in Figure 7. Thus, we can integrate the models by providing categorical bridges from one type of specification to another. A DeMarco specification, for example, can be mapped to a number of Yourdon specifications via a structured task. The system designer can then exercise discretion in choosing the appropriate option according to system requirements.

![Diagram](image)

Figure 7. Categorical Relationships among Structured Models

7. CONCLUSION

A category-theoretic approach is proposed in this paper to link up the structured analysis and design models. The concepts of structured tasks and morphisms are defined. They help a system developer visualize the internal structure of a system, assemble or refine subsystems, and verify the consistency and completeness of a design. These can be done through simple set-manipulation with the aid of task diagrams.

The structured tasks and the morphisms form a category. Similar categories can be defined over other structured models such as DeMarco data flow diagrams, Yourdon structure chart and Jackson structured text. We can integrate the models by providing categorical bridges from one type of specification to another via the structured tasks. A DeMarco specification, for example, can be mapped to a number of Yourdon specifications. The system developer will be presented with possible design choices and may select the appropriate implementation based on personal experience, environmental considerations and hardware characteristics.

ACKNOWLEDGEMENTS

Part of this research was done at the London School of Economics, University of London under a Commonwealth Academic Staff Scholarship. The author is grateful to Mr Ronald Stamper, Dr Haya Freedman and Professor Frank Land of LSE, Dr Kit Grindley of Price Waterhouse and Dr Sami Zahran of ICL (UK) for their invaluable comments and suggestions. The research was also supported in part by a University of Hong Kong Research Grant.

References


CALL FOR PAPERS FOR A SPECIAL ISSUE ON
DIGITAL IMAGE PROCESSING

The Australian Computer Journal will publish a special issue on 'Digital Image Processing' in August, 1987, under the guest editorship of John O'Callaghan of the CSIRO Division of Information Technology in Canberra. Research papers, tutorial articles and industry case studies on any aspects of the subject including computer vision, hand-eye systems, image restoration and reconstruction, image enhancement, image compression or image analysis will be welcome. Both full papers and short communications will be considered and prospective authors should write to, or telephone, as soon as possible:

Dr J.F. O'Callaghan,
CSIRO Division of Information Technology,
PO Box 1800,
Canberra City, ACT 2601.
(Telephone (06) 433-204)

to inform him of their intention to submit material for the issue and to provide a brief summary of their proposed contribution. In order to allow adequate time for refereeing and editorial review, complete manuscripts will be required by 20 March, 1987. These should be prepared in accordance with the guidelines published in the February, 1986 issue of this Journal.


Biographical Note

T.H. Tse received his B.Sc. degree from the University of Hong Kong in 1970 and his M.Sc. degree from the University of London in 1979. He is currently a lecturer in computer science at the University of Hong Kong. His research interests include formal methods in information systems and software engineering.

Mr Tse is a member of the British Computer Society, the British Institute of Management and the Institute of Data Processing Management. He is a council member of the Vocational Training Council in Hong Kong. He was awarded an M.B.E. by the Queen in 1982.
The Use of Non-hierarchical Allocation Methods for Clustering Large Sets of Data

L. Belbin

Requirements for the pattern analysis of increasingly large sets of data have elevated the utility of non-hierarchical methods. While SAHN (hierarchical) techniques are still very much in vogue, they are unable to analyse large sets of data on fast computers, and in many cases represent an inappropriate strategy. A simple allocation algorithm with an enhanced method for representing association is presented and compared with a standard SAHN methodology. Results suggest that non-hierarchical techniques have been under-estimated.

Keywords and Phrases: Cluster analysis, data processing, numerical taxonomy, large datasets.

INTRODUCTION

The aim of this paper is to present a case for more detailed investigation of non-hierarchical (NH) algorithms in pattern analysis. While a simple algorithm is presented to illustrate the utility of the method in general, the intention is to compare and contrast this simple method with what is suggested as the most widely used hierarchical technique. Features of the different NH algorithms, such as the different approaches to the handling of mixed attributes (nominal, ordinal, interval and ratio) and the complexities of 'seeding' make such comparisons a separate issue. A review of some of the highlights of NH algorithms is presented and key features are detailed in relation to the method used in this paper. In this regard, the method presented is built on what have been shown to be useful features of previous methods as well as a blending in of a new approach to mixed data. The emphasis is placed on the marrying of an appropriate technique to the specifications of the data and the requirements of the analyst.

The most widely used techniques in Numerical Taxonomy or Pattern Analysis (Blashfield and Alderneuer, 1978) are those clustering methods termed SAHN or Sequential-Hierarchical-Agglomerative-Non Overlapping (Sneath and Sokal, 1973). The analyst begins with a set of m attribute values for each of n objects. In the SAHN approach, the objects are initially recorded as n distinct groups, each group comprising a single object. In the SAHN approach, the products include a list of the fusion history; which objects or groups fused, and at what level of association.

Non-hierarchical methods begin with some sub-optimal partitioning of objects into groups and iterate by moving objects from one group to another so as to optimise some measurement of within-group homogeneity and between-group heterogeneity. These methods are designed to produce a partitioning of objects into a set of groups but give no direct information as to how the groups, or the individuals within them are related. The aim of this paper is to advance a simple NH algorithm that provides results comparable with SAHN methods but is applicable in sets of data 100 times larger than the latter techniques can generally handle.

While the aim of NH clustering methods is to produce some optimal partitioning of objects into clusters, SAHN methods are designed to optimise the fusions and therefore define clusters only in an indirect fashion. Most tend to use SAHN methods to provide a partitioning, making little or no use of the inter-group structure. The neglect or avoidance of the NH methods under these circumstances is surprising and can be put down largely to two aspects of their procedure: they do not produce an hierarchical structure and they do require some form of initial configuration. Perhaps because of these limitations, the NH methods have been assumed, I believe incorrectly, to be inferior.

While both SAHN and NH methods are iterative and essentially simple in concept, they are initiated by different assumptions, proceed by optimising different criteria and therefore arrive at a different goal. SAHN algorithms begin by utilising a symmetric matrix of associations or proximities between all pairs of objects. While a dataset may have contained 1,500 objects and 500 attributes or 750,000 values, the matrix of associations between all

Copyright ©1987, Australian Computer Society Inc.

General permission to republish, but not for profit, all or part of this material is granted, provided that the AJC's copyright notice is given and that reference is made to the publication, its date of issue, and to the fact that reprinting privileges were granted by permission of the Australian Computer Society Inc.

1 Division of Water and Land Resources CSIRO, GPO Box 1666, Canberra, ACT, 2601. Manuscript received June, 1986; revised November, 1986.

pairs of objects contains \(1500^2 \times (1500-1)/2\) or 1,124,250 entries. NH methods on the other hand, replace the inter-object associations which are usually resident in computer memory with an object to group association that does not need to be retained in memory. Even the data need not be memory resident. These features provide NH algorithms with a considerable computational advantage. So while SAHN algorithms remain the most popular, they are amongst the most computational and memory intensive algorithms to be found in Numerical Taxonomy. Under such circumstances, it is fair to say that they remain popular because people who use them consider that they generally provide good classifications.

The drawback often alluded to with NH methods is the requirement for the seemingly subjective prescription of the number and nature of the initial partitions or seed objects (points). Various solutions to this problem have led to a variety of allocation strategies.

The initial clusters may be defined by some partitioning of all objects in the data set, or by the use of a subset of objects as group centroids or by the use of some type of derived or artificial seeds. MacQueen (1967) chose the first \(k\) objects in the dataset as seed points, Anderberg (1973) suggested extracting seeds on regular intervals so as to reduce any original sampling bias, while McRae (1971) and Gauch (1979) used a random number generator to select data objects as seeds. Beal (1969a, b) set up artificial seed points using one standard deviation separation on each attribute while Thorndike (1953) selected a set of data points that were mutually furthest apart. Anderberg (1973) also suggested the partitioning of the attributes into classes to derive a set of synthetic objects that attempted to evenly cover the attribute space. Ball and Hall (1967) determined the grand centroid and scanned the dataset, selecting objects greater than some threshold value away from the seeds already defined. Astrahan (1970) attempted to get around the problem of good coverage of the attribute space by ordering the objects by the decreasing density of the surrounding objects in their neighbourhoods and selecting sufficient seeds so as to have a high probability of coverage of the complete attribute space. Lance and Williams (1975) provided a novel algorithm (REMUL) which side-stepped the seeding problem. Aspects of this algorithm such as the use of the Canberra Metric and the removal of objects from groups before calculation of object-group distance are included in the modules DIPCAN, RECAN and AXCAN of the package TAXON (see Ross, 1984). This method involved alternating between monothetic division (using a single attribute at each split to divide a group) producing two daughter groups and then permitting re-allocation of the objects between the groups based on the Canberra Metric dissimilarity measure.

Some methods attempted to circumvent problems associated with seed generation or selection by strengthening subsequent phrases. For example, the second phase of many NH algorithms provides for the allocation of all objects to groups, each group being based on one of the initial seed points, followed by a re-calculation of group-centres. The objects are usually selected sequentially and the distance (using some measure of association) from object to all group centres, represented by the seed points, is examined. The minimal value indicates the preferred group for that object. Should the allocation of a new object to a group alter the definition (usually the centroid) of that group? Forgy (1965) initially allocates all objects to a set of centroids that are not updated by addition. This avoids any bias due to ordering in the data. After all objects are allocated, the centroids of the groups are re-calculated and subsequently updated by addition. Gauch (1979), as noted, selects seeds randomly and assigns objects to these fixed centres, but only those within a user defined radius. In this case, the random seed selection and fixed allocation are alternated, allocating only those objects yet un-classified. MacQueen (1967) updates the centroids with the addition of each data point leading to sequence dependent partitions. Lance and Williams (1967b) suggested using a suitable SAHN method on a subset of the data to produce clusters and thereby obtain a set of group centroids.

One of the referees has pointed out a significant feature in relation to the use of centroids for the calculation of distances. While most NH algorithms use the centroid as a summary, or substitute for the members of a group, the SAHN algorithms that support such a feature have been actively avoided. The reason for not using the centroid strategies in hierarchical methods has been based on the fact that they are likely to produce what are termed reversals in the accepted monotonic increase in fusion distance (normally the distance increases for each successive fusion). The justification for the use of centroids in non-hierarchical methods is simple, their use provides for a significant reduction in computational time without any parallel drawbacks. Time is reduced from a value proportional to \(n(n-1)/2\) to a value proportional to \(n.G\), where \(n\) = number of objects, when a group is defined in the algorithm as being represented by each of its members, to a value \(n.G\), where \(G\) = number of groups, when the group is represented by its centroid.

Alternative heuristics for the progression of the initial partitions to some type of optimal configuration amount to variants on the choice of how and when group centroids are to be updated. Forgy (1965) and Lance and Williams (1975) update the centroids at the completion of each iteration (the determination of closest group for each object). As noted above, MacQueen, by comparison, updates group centroids with the allocation of each object. His approach also contrasts to that of Forgy in that the last iteration reverts to fixed centroids. Jancey (1966) suggested that, on the allocation of an object to a group, the direction of motion of the old to new centroid could be used to speed convergence by the use of a small additional movement.

Enhancements to the basic allocation algorithms go from the simple to the complex. Gauch (1979) for example, recognising that his COMCLUS algorithm is likely to leave regions between major groups (interstices) containing a small number of objects, re-allocates them to the nearest large cluster. A more complex approach includes the use of Coarsening (C) and Refining (R) parameters in an
attempt to better adapt to the natural number of clusters in the data. MacQueen (1967) chooses the first k data units as seed points and computes all the pairwise distances between seeds. Any seeds closer than the user defined C parameter are merged and inter-seed distances re-calculated. When all inter-seed distances are greater than C, allocation of all remaining data units to seeds proceeds. If the updated centroid is less than C to its nearest neighbour centroid, the two groups are fused together. If the distance of the object to nearest centroid is greater than the Refining parameter R, then the object seeds a new group by itself.

Unlike MacQueen's algorithm which stops after the first re-location, Wishart's RELOCATE in CLUSTAN (1978) iterates until convergence and adds further refinements. A bias of an object to its currently assigned group is (optionally) countered by removing it before distances to all centroids are calculated. This approach was also used by Lance and Williams (1975) in REMUL, and subsequent algorithms based on it in the TAXON package (see Ross, 1983). RELOCATE uses a refining threshold such that a data unit showing a distance to its closest centroid greater than this value is allocated to a residue group. A minimum cluster size is also defined and acts in the same way; clusters containing less than this number of data units are disbanded to the residue group. Another feature of RELOCATE is the calculation in inter-group distances and the fusion of the two closest groups. The iteration phase is then re-commenced and the process continues until the parameter nominating the minimum number of clusters is achieved.

Further elaboration is found in ISODATA (Ball and Hall, 1965) and its derivatives (Ball and Hall, 1967; Sammon, 1968; Wolf, 1968; Hall et al., 1969; Dubes, 1970; Wishart, 1978). In common with previous methods, centroids are updated after complete allocation of objects to groups. Groups with less than a user defined number of objects are disbanded and inter-seed distances recalculated. When all inter-seed distances are greater than C, allocation of all remaining data units to seeds proceeds. If the updated centroid is less than C to its nearest neighbour centroid, the two groups are fused together. If the distance of the object to nearest centroid is greater than the Refining parameter R, then the object seeds a new group by itself.

The first phase of ALOC consists of calculating the first re-location, Wishart's RELOCATE in CLUSTAN (1978) iterates until convergence and adds further refinements. A bias of an object to its currently assigned group is (optionally) countered by removing it before distances to all centroids are calculated. This approach was also used by Lance and Williams (1975) in REMUL, and subsequent algorithms based on it in the TAXON package (see Ross, 1983). RELOCATE uses a refining threshold such that a data unit showing a distance to its closest centroid greater than this value is allocated to a residue group. A minimum cluster size is also defined and acts in the same way; clusters containing less than this number of data units are disbanded to the residue group. Another feature of RELOCATE is the calculation in inter-group distances and the fusion of the two closest groups. The iteration phase is then re-commenced and the process continues until the parameter nominating the minimum number of clusters is achieved.

THE PROPOSED ALGORITHM

The overall simplicity of the non-hierarchical allocation strategies and their ability to accommodate very large datasets in near linear time is appealing. Comparative studies (Milligan, 1979) suggested that these techniques were under-rated when compared with the more widely used algorithms of SAHN type. The problem was to select from among the variety of features of NH algorithms those that imparted significant gains to a variety of data with algorithmic simplicity and efficiency. Experience has suggested that the simpler the algorithm, the more interpretable the results. The algorithm presented here (ALOC), has been incorporated into the Pattern Analysis Package (PATN) developed at the Division of Water and Land Resources, CSIRO (Belbin et al., 1984).

The first problem to address was the generation of some starting configuration. The methods reviewed had the drawback of depending on one or at most a few methods of generating the initial configuration. A suitable algorithm often combined a poor start with a good iterative algorithm or vice versa. ALOC on the other hand, expects a set of seed points to have been pre-defined. Instead of avoiding the problem, the intention was to allow the analyst to use other facilities already existing in PATN to provide information appropriate for their purpose.

This study suggested that an ability to overcome a poor initial configuration was more useful than attempting to find one foolproof method of generating an initial configuration that was independent of data structure. ALOC appears to be able to produce excellent results from very poor seed configurations. Another factor relating to starting configurations is that as the sophistication of the method for generating the initial configuration increases, so does the requirement for computing resources. Considering that this type of algorithm is vital for the analysis of large sets of data, this additional time can be a significant drawback. ALOC uses MacQueen's (1967) concept of a threshold distance to create additional seed points. This feature, in conjunction with the other components of the algorithm has provided good solutions, robust to a range of data structures.

The first phase of ALOC consists of calculating the distance of each object to all needs. If, for any one object, the minimum of these distances is greater than the user supplied threshold, that object becomes an additional seed and the examination continues. If the number of initial seeds and the threshold distance is small, the method approaches other techniques that attempt to cover the
attribute space with evenly distributed seed points.

The second phase comprises a single pass of all objects to fixed seed points that have been entered by the user or generated by the first phase. Fixed, in this sense means that the seed point (centroid) is not influenced by the addition of objects. This is used to gain independence of the ordering of objects in the data.

The next step in the algorithm is a re-estimation of the group centroids, excluding the seed points. The exclusion implies that while the initial seeding may be comprised of artificial objects, the main part of the algorithm will be defined only in terms of the objects comprising the dataset. From this phase on, the object-group distances and the group-group distances are defined in terms of the group centroid.

The iterative phase is defined by taking each object in turn, calculating the closest group and if necessary, re-allocating it to a closer group. Two details are important. First, before the object-group distances are estimated, the object is removed from the group to which it currently belongs. This simply avoids the bias of weighting the centroid toward the object under examination. The second feature is the immediate re-estimation of the centroid of the group to which the object has been assigned, giving equal weight to all objects in the group.

Two direct stopping rules are provided: the maximum number of iterations and a minimum number of re-allocations. The former is intended to be data independent and to provide a facility for controlling the iterative phase from zero iterations (fixed allocation with no re-allocations) to unconstrained. The latter is more often used so as to allow the algorithm full opportunity to converge to the point where all objects have been allocated to the group with the closest centroid. Convergence can however only be guaranteed (Selim and Ismail, 1984) for metrics in a Euclidean space. Four years of experience with the algorithms using a variety of complex non-metric association measures has disclosed no divergent situations (the number of re-allocations during a single iteration either continues to increase or to oscillate).

Rather than attempt to build any criteria for subsequent merging and splitting of the resulting groups into the algorithm, the strategy adopted results in an inter-group distance matrix that may be used for subsequent analysis by more suitable techniques for displaying such relationships. As noted above, the SAHN techniques are appropriate for this; the pertinent techniques for optimal partitioning and inter-group structure are utilised. The use of internal criteria for group merging or splitting such as in ISODATA may result in the fusion of two groups that have little or nothing in common. The ability of the algorithm to create as many groups as necessary in the first pass is balanced by subsequent analysis of the inter-group distance matrix; no fusion is forced. It is then up to the analyst through the examination of the groups and inter-group structure to decide if a suitable number of groups has been forthcoming. If there remains considerable heterogeneity within one or more final groups, then insufficient groups have been created. ALOC will need to be re-run, either by allowing it to generate more groups by use of a smaller threshold distance or by the user including more initial seeds. The inter-group structure can be conveniently examined by a SAHN algorithm. If some of the groups warrant fusing, it can be done at this stage. The ability of the non-hierarchical method to create good partitions should not be compromised by the imposition of optimising criteria from a strategy with a different goal.

The inter-group association matrix is generated in the same way as the object to group associations, using the selected combinations of association measures and weighting for the set of attributes as detailed below. This can be achieved because groups are represented by centroids and are therefore equivalent (though not necessarily identical) to an object, maintaining the same units for object-group and group-group distances.

A few situations have been observed where oscillation of a set of objects occurs between a number of groups. This should not be confused with divergence. While divergence indicates instability in the algorithm in relation to a set of data, oscillation depicts situations where affinity of objects of two groups is equivocal. For example, oscillation can occur when a configuration such as seen in Figure 1 is generated. In this situation, object 1 may be marginally closer to group 2 than group 1, so an exchange occurs. Now that object 1 has been removed from group 1, object 2 becomes closer to group 2 than group 1, and so on. Now that object 1 has been removed from group 1, object 2 becomes closer to group 3 and therefore joins this group. Now, however, because of the removal of object 2 from group 1, object 1 is again closer to group 1 and moves back into that group. The updated group 1 centre now favours the inclusion of object 2 from group 3 tipping the balance for the oscillation to begin again. This type of oscillation is detected, and the configuration producing minimal heterogeneity is accepted as final.
THE DETERMINATION OF OBJECT-GROUP ASSOCIATIONS

In the foregoing discussion nothing has been said concerning details of the method used to calculate the association between objects and groups or finally, between groups. Most of the original techniques assumed or utilised Euclidean distance. Recent evidence (Faith et al., in press) suggests that there should be no aversion to using non-metric associations. In this study non-metrics appeared to provide superior performance in quantifying affinity. Although this study was in an ecological context, it can be assumed that the same conclusion may apply to other applications using similar data.

Weir (1970) suggested a novel approach whereby true mixed data (data with combinations of NOMINAL, ORDINAL, INTERVAL and RATIO attributes: see Anderberg, 1973) could be effectively handled. The concept allows for the application of a suitable measure of association and grouping, independently for each attribute. ALOC uses such an approach.

The association measures provided by the algorithm include Bray and Curtis (1957), Canberra Metric (Lance and Williams, 1967a), Minkowski series (see Sneath and Sokal, 1973), Gower metric (Gower, 1971) and a version of the Gower metric suitable for handling data measured as compass bearings. When applied to presence/absence (1/0) data the first two measures correspond to Czekanowski (1913) and Jaccard (1901) respectively and the last three to the Simple Matching Coefficient (see Clifford and Stephenson, 1975).

The ability to group attributes encourages the establishment of equal weighting to suites of attributes where appropriate, as well as the re-establishment of a more intuitive weighting after recoding of non-ratio type attributes to ratio type. The former is useful if the suites represent different logical aspects of the data. For example, sites may have been measured for geology, soil, vegetation and mammals and it may be desired to maintain the equality of weight between each of these sets, regardless of the number of attributes in each. Re-estimation of the latter weighting may be required when, for example, colour, originally on a nominal scale 1-20 is transformed into 20 new presence/absence type attributes such as green/not green, red/not red. In this case, the new attributes will have 20 times their original weighting.

EXAMPLES

An artificial dataset was created using the RAND option of the PATN package so as to contain various structures in a two-dimensional space nominally 100 units in the x-direction and 100 units in the y-direction (Figure 2). A two-dimensional test set was chosen as the foundation of the tests primarily for purposes of display, however a higher dimensional derivation of this same test set suggested the utility of the algorithm did not deteriorate with higher data dimensionality.

The dataset of 400 objects was built up in six stages (see Figure 3). Three groups of 50 objects (sequence numbers: 1-50, 51-100, 101-150) each were generated using bivariate normal distributions and the location of the means were placed so as to produce a north-east trend of groups with decreasing radii, starting at the centre of the space. Moving from the centre toward the north-east quadrant, the three groups had standard deviations of 6, 4 and 2 units respectively. A fourth group (151-250) located in the south-west quadrant was generated using a bivariate uniform distribution and was localised to the south-east quadrant. A sixth group (351-400) was then generated with a bi-variate uniform distribution...
Non-hierarchical Allocation Methods for Clustering

A standard analysis using Euclidean distance in the two dimensions and the UPGMA SAHN method (Unweighted Pair Group Arithmetic Averaging, see Sneath and Sokal) was performed. The five group level was chosen for summary and is displayed in Figure 3. This methodology recovered five of the six groups as generated. The notable difference was the lack of discrimination between two and three in the north-east quadrant. In addition, a new group comprising objects from group six was established. Considering the size and spread of the other three groups, and the isolation of 10 objects of group six in the north-west quadrant of the sample space, the UPGMA classification presents a reasonable solution.

ALOC also used Euclidean distance, while three different strategies were chosen to test the robustness of the algorithm to various starting configurations. A threshold radius of 25 units (one quarter of the distance of one side of the two-dimensional space) was used. New seeds would form if the minimum distance of an object to all existing groups exceeded this value. For purposes of comparison, the inter-group association matrix resulting from ALOC was used to fuse the number of groups to five using the UPGMA algorithm.

The first configuration used a random selection of 20 of the 400 objects as seeds (20 groups). Use of the allocation radius of 25 units resulted in 25 groups. Subsequent fusion as noted above to the five group level resulted in the configuration as shown in Figure 4. There is minimal difference from the UPGMA result. Of the 400 objects, only five have been classified into different groups; objects 368, 373, 385, 397 and 400. All these objects could be said to be on the fringe of the central group while their affinities are ambivalent between adjacent groups. For example, object 373 on the southern edge of the central group was classified as part of the south-western group by UPGMA but ALOC places it, more reasonably in the central group. The same can be said of the other four objects; examination of each leads to a conclusion that there is little to distinguish between the two partitions.

The second method used 20 objects all selected from the extreme south-west corner of the sample space. These were selected on the basis that they represented hypothetically, one of the worst-case situations for seeding ALOC. The 25 unit threshold resulted in ALOC generating an additional five groups (giving 25) that fused as noted to the five group level (Figure 5). Only four objects; 357, 366, 368 and 400 were different to the UPGMA result. All, as for the random start, were marginal in the sense that they were located midway between more clearly defined groups. Notably, object 400 on the western side of the central group is probably better located in the western group in this classification rather than in the central group as with UPGMA.

The third starting configuration used only the first object as a seed group and resulted in eight additional groups being formed. Again, the result (Figure 6) is essentially congruent with UPGMA and the other two ALOC classifications. The same four objects as the 'bad' start were classified differently to UPGMA while the only difference with the previous classification was the allocation of object 400 to the north-western group. Over the past three years, this simple method of seeding ALOC has been found to be ideal.

Of the three different ALOC starting configurations, a maximum of five out of the 400 objects (1.25%) differed from the UPGMA classification. While not suggesting that the UPGMA grouping is optimal, visual evaluation of all classifications would suggest that ALOC, even given an extremely poor starting configuration, performed well; any of the resulting classifications could be regarded as providing a very good recovery of the data structure.
Non-hierarchical Allocation Methods for Clustering

this can be said to be at least partly dependent given the ideal Euclidean distances, there is no reason to suggest that ALOC would perform any worse than say UPGMA because of the introduction of poorly estimated distances or a higher dimensionality.

To pursue this point, an alternate version of the test set was generated and analysed. The COEN algorithm (Minchin, 1983; Belbin et al., 1984) was used to produce a set of 24 attributes distributed over the same two dimensional (x—y) space as used above. Each attribute can be thought of as a three dimensional surface generated by a beta function (see Minchin, 1983). The beta function was used because it can generate surfaces varying from near normal to highly skewed. The 24 surfaces were generated with randomly varying means (x and y) and amplitudes (z). The amplitudes of these surfaces were sampled at the same co-ordinates as the two-dimensional set so as to produce what would appear superficially as a 24-dimensional set of data, but in reality 24 attributes representing data in a two-dimensional plane. It was hoped that ALOC would be able to reconstruct the same or a similar classification of these data as produced by the two-dimensional data.

The multi-dimensional scaling program KYST (Kruskal et al., 1973) was used on the Bray and Curtis association measure to provide a two-dimensional base for subsequent classifications. This program accepts a matrix of similarities or dissimilarities representing some type of affinity between objects and attempts, for a given (chosen) dimensionality, to find a distribution of these objects such that the distances between them in the reduced dimensional space, matches the input distances most closely. The association measure of Bray and Curtis (1957) was used. Simulation studies (Faith et al., in press) suggest that this measure is appropriate for this type of data. Figure 7 is the result after applying KYST to these data and requesting a two-dimensional solution. Although some rotation of the objects from their original positions in two-dimensions is evident and the peripheral groups have been distributed toward the circumference, original structures are recognisable.

The UPGMA classification based on the 24 attributes and using the Bray and Curtis measure is displayed at the five group level in Figure 8 (using Figure 7 as a base). This shows the formation of a larger western group comprising the previous north-west and south-west groups. The additional group comes from the partitioning of a southern group from the previous south-western group. This solution is an acceptable alternative to the two-dimensional classification given the two-dimensional view of the data as seen via the Bray and Curtis measure (Figure 7).

The same procedure for evaluating ALOC for the two-dimensional set of data was now applied to the 24 attribute set. Each of the three starting configurations for ALOC produced a classification almost identical to each other and to the UPGMA result. Only two of the 400 samples (357 and 373) were allocated in a different manner from the UPGMA result. Again, these were two of the intermediate samples as in the two-dimensional result, intermediate in position between two group centres. Figure 9, using Figure 7 as a base, shows the ALOC grouping using a single seed and a threshold of 0.3 for the Bray and Curtis measure (ranging from zero distance to unity).

In addition to the artificial sets described above, ALOC was used to classify a dataset (Day and Nix, in prep.) of 14,308 samples of climate from a 1/10 by 1/10° grid covering northern Australia (north of 20°S). This set was included as an example analysis of a large set of data that resulted in smooth and interpretable patterns. Twelve variables (Table 1) comprising rainfall and temperature measurements were derived from a climatic database using surface fitting techniques (Hutchinson, 1984; Nix et al., in prep.). For this analysis, two samples were used as seed
points, the extreme south-west site and the site at the extreme north-east (Cape York). A threshold value of the Gower metric of 0.05 was used to generate 35 additional groups (any object having a minimal value of the Gower metric greater than 0.05 will form a new group). The result is displayed as Figure 10 with colours allocated using the automatic colouring algorithm (Belbin et al., 1984). This algorithm attempts to portray differences between the groups in terms of colour differences.

The examination of Figure 10 suggests that ALOC has provided an excellent display of the climatic variation of northern Australia. The overall trend away from the coast representing the diminishing rainfall and a decrease in minimum temperatures is obvious. Features such as the highlighting of the higher plateau of the Pilbara region of the north-west is meaningful. This region is identified by being hotter in the driest quarter and generally drier than surrounding regions. Another notable region, located just south of Cairns (blue in colour) is unique with its lower temperatures and extreme rainfall. Bartle Frere, a mountain in this region is 1622 m and receives over 3800 mm rainfall a year, one of the highest recordings in Australia. The region south-east of Townsville (brown area on the south-east margin of the region) stands out as having lower rainfall than surrounding regions due to a rain shadow effect from the south-east.

Considerable detail is provided by combining the classification using ALOC and the automatic colouring algorithm. In common with most pattern analysis studies, it not only shows overall trends, but highlights interesting detail promoting further questioning of the database as well as the methods used to generate the data and in this case the physical models involved.

**DISCUSSION**

While allocation methods appear capable of partitioning sets of data of the order of tens of thousands of objects into homogenous clusters, little use appears to have been made of them in any discipline in comparison with SAHN strategies. This appears due largely to the question over the selection of an appropriate partition of the dataset and possibly the assumption that only more complicated techniques can produce meaningful results.

The fact that SAHN techniques are most often used to produce a partitioning of the data rather than their forte, the production of an optimal hierarchy, is often overlooked. The dendrogram resulting from an hierarchical analysis usually contains more information than can be comfortably assimilated. Generally, a straight line is drawn on the dendrogram at a subjectively selected level of dissimilarity to define a set of groups to be examined in more detail. The hierarchical techniques are therefore more often used to partition than to provide details of inter-group relationships. By comparison, the non-hierarchical methods attempt to optimise the group structure without regard to inter-group relationships.

---

**Table 1. The Twelve Measured Climatic Attributes Used in the Climatic Analysis of Northern Australia.**

<table>
<thead>
<tr>
<th>Temperature:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annual mean</td>
</tr>
<tr>
<td>Minimum monthly</td>
</tr>
<tr>
<td>Maximum monthly</td>
</tr>
<tr>
<td>Range</td>
</tr>
<tr>
<td>Wettest quarter</td>
</tr>
<tr>
<td>Driest quarter</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rainfall:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annual total</td>
</tr>
<tr>
<td>Minimum monthly</td>
</tr>
<tr>
<td>Maximum monthly</td>
</tr>
<tr>
<td>Range</td>
</tr>
<tr>
<td>Wettest quarter</td>
</tr>
<tr>
<td>Driest quarter</td>
</tr>
</tbody>
</table>
Some would suggest that the imposition of a hierarchical structure upon data that were not essentially hierarchical is a fundamental error. It is notable that many workers in pattern analysis have in fact assumed that chaining, the successive fusion of single objects to a group, was a condition to be circumvented by selecting appropriate fusion strategies (Belbin, 1984) rather than evidence of any non-hierarchical data structure. A further problem evidenced by SAHN techniques, and especially those termed space dilating (Belbin, 1984; Abel and Williams, 1985) is that the analyst often falsely assumes that well separated clusters are evidenced by the dendogram. What is more often the case is that while two group centroids may be well separated, their group boundaries may be overlapping.

The allocation strategies make fewer demands in that they impose a partition rather than a hierarchy; they also make fewer computational demands because comparisons are limited to those between objects and centroids. The former is evidenced in that a hierarchy assumes both a set of partitions and an inter-partitional structure. It therefore contains more information than a partitional structure alone. The cost of the additional information is borne by an additional computational load.

Where does ALOC stand on the question of optimality? There is no doubt that this algorithm is unlikely to achieve a global optimum in the sense of finding an arrangement of objects into groups that show maximal within group homogeneity and maximal between group heterogeneity. Even for a fixed (known) number of groups, the task of finding this global optimum exceeds most computational limits, there being of the order of $5^{15}$ arrangements of 15 objects given five groups. If a single seed and a reasonable threshold are provided, the algorithm does however aim at this arrangement. The generated seeds will establish the initial between-group heterogeneity and the allocation-reallocation phase attempts to maximise within-group homogeneity. The stopping-rule 'iterate until no object can be moved to a closer group (than the one it is in)' is the heuristic or rule of thumb that determines optimality.

Other forms of optimality criteria have been used for similar type algorithms. Direct minimisation of error sum-of-squares or variance were discarded as inappropriate for data where knowledge of the attribute statistical distribution parameters were unknown. ALOC maintains, as a measure of heterogeneity, the sum of $n$ distances of each object to the group to which it is currently assigned. Experience suggests that using its current criteria of optimality (iterate to stability), ALOC provides opportunity for the algorithm to break out of local optimum as measured in terms of the heterogeneity. In summary, it is expected ALOC, as a heuristic, provides a good approximation to the global optima of homogeneous groups.

There are two order-dependencies in the algorithm; the generation of additional seeds at phase 1 and the movement of objects between groups in phase 3. The ordering of the data determines which objects form additional seeds because each object is compared with all existing (prior) seed points. The updating rule is also order-dependent in that the relocation of an object to a different group is immediate and, seeing that each iteration scans the data in sequence, dependent on the initial ordering of the data. These features enhance the speed of the algorithm at the expense of forgoing alternative starting and intermediate configurations.

The technique of allocation lends itself to analysis of additional samples at some later date. The centroids from a previous NH analysis form the seeds while the new objects form the data. For example, a comprehensive study of a region may be followed up by the allocation of sites subsequently measured from similar regions to the original management groups.

Experience suggests it is computationally expedient to partition large sets of data into a greater number of groups than desired for final summary, and then use hierarchical fusion or ordination techniques (suggest multidimensional scaling) to analyse the inter-group relationships.

REFERENCES
CALL FOR PAPERS

18th Annual CAE Computer Conference
Adelaide, 24-26 August 1987

The aim of the conference is to examine the maturity of CAE computing after 18 years. Papers are invited on any aspect of CAE research, teaching or administration. Intending authors should send a brief abstract by 1 March to:

18th Annual CAE Conference, Mathematics and Computing Studies, SA Institute of Technology, PO Box 1, Ingle Farm, SA, 5098
Telephone (08) 343-3343


41
Book Reviews


As its name implies, this is really a book for the EDP auditor, but it contains lessons that all data base and application system designers should take to heart. The book is concerned with defective data bases, means of checking their integrity, and methods for preventing defects from creeping in. Integrity Analysis is a first step in providing:
— proven models to disclose and analyse data defects and to monitor data base integrity
— a formal approach to data verification
— facilities for data repair to support both operations and quality assurance
— standards and guidelines for application design to anticipate and deal with defects in data used to drive program logic.

The author's studies support the fact of defective data bases, some of the defects no doubt carried over from earlier systems. Her discussion of the consequences of defective data are tied to examples of methods of finding defects for correction. It constitutes a salutary lesson for all who have designed applications, in particular the editing of the input data. Her guidelines for preventing erroneous data entry appear to be common sense, but how many of us have really thought the problem through?

A large portion of the book is concerned with sampling techniques for large data bases, to allow auditors to check the integrity of the records. But there are clear guidelines too, for application designers. These take the form of constraint specifications for checking of data, and suggestions for documenting them. The aim is to ensure correct data input through internal consistency of records, as well as through the more common overall hash totals and the like.

Lest one be daunted by the detail given in the general chapters, there is a clear case study and an appendix giving guidelines for data verification, so that the designer can:
— establish the integrity of the input data
— minimise the subsequent injection of defects
— disclose and monitor data integrity violations
— anticipate and handle as exceptions any defects that appear in the course of processing.

In all, not an easy book to read, but well worth the effort.

Des Bright
IBM Australia


The book covers systems development. It concentrates on the linear cycle made up of five phases, namely, preliminary investigation, analysis, design, which is divided into preliminary and detailed design, development and implementation. Before describing these phases the book spends two chapters outlining the roles of systems analysis and the skills needed by them.

Preliminary investigation concentrates on problem definition and objective setting but has very little on cost-benefit analysis, which is left to the chapters on preliminary design.

Systems analysis begins by describing information gathering techniques such as questionnaires and interviews. It then describes system modelling using structured systems analysis. The emphasis here is on processes. Data modelling receives very little attention.

System design is divided into two parts, preliminary design and detailed design. Some ideas about prototyping are introduced in preliminary design. Detailed design covers many topics, including input and output design, processing of files and database design. After describing system development and system implementation the book concludes with chapters on system controls, including project management, system security and some contemporary trends, in particular use of personal computers, office automation and the information centre.

In summary the book is much stronger on design than analysis and does not cover analysis techniques to any great depth. It concentrates on processes and does not cover some new ideas such as conceptual modelling or relational analysis, which takes about two pages in the book. Many design techniques are introduced in the book but none is developed to any great depth.

Each chapter concludes with a set of review questions and case studies. The review questions are usually of a discussion nature, whereas cases are broad in nature rather than technique oriented. Thus students would either need assistance in the application of the techniques to the cases or alternatively some more elementary exercises would have to be set by a lecturer to illustrate techniques described in the text.

I.T. Hawryszkiewycz
New South Wales Institute of Technology


This book is the fifth in the Springer Books in Professional Computing series. The authors indicate that it came into existence as a case studies report written at SofTech Inc, as part of an effort to identify and resolve issues related to Ada usage. Rewritten into a textbook it serves a wider purpose. At one level it can be used as a source book in software engineering. It discusses a range of specific and structural issues that arise in software engineering projects and uses as a specific example a nontrivial message-switch problem in a communications application. The issues are discussed in general terms, and solutions given in the Ada language.

At another level it can be used as a source book for good programming practice in Ada, targeted to competent programmers who are not familiar with Ada.

As befits a software engineering book it can either be read bottom-up (which happens to correspend to forwards) or top-down (backwards).

Chapter One addresses the problem of selecting identifiers for the wide range of entities that occur in a nontrivial problem. This problem is exacerbated in a language with the package/module construct where an identifier can represent an imported entity and effectively has to provide a description of the entity, without the benefit of a local declaration. A naming convention is proposed which, of course, is applicable to all sensible programming languages. Chapter Two discusses the type abstraction in general and the definition of types in Ada in particular. This material will be familiar to programmers trained in Pascal/Modula 2.

Chapter Three discusses a range of coding paradigms and their implementation in terms of Ada constructs (slices, short-circuit control forms, and loop constructs). Chapter Four discusses aspects of the exception mechanism provided by Ada. A range of applications of this mechanism is given, ranging from the implementation of a control structure to the handling of a detected software error. In the message-switch problem this corresponds to the case where a message is accepted by one validation module and is subsequently rejected by another (independent) validation module.

The remaining two chapters address broader issues. Chapter Five is concerned with aspects of building a program out of separate subunits. In particular the chapter discusses the goal of designing reusable software, information hiding and its relation to data abstraction, and the spectrum between the top-down and the bottom-up factoring of a program into subunits. Finally Chapter Six analyses the impact that a "wide-spectrum" language such as Ada can have on the classical view of the software life cycle.

There is obviously a great deal of developed technique and accumulated wisdom in professional software houses. The software engineering discipline will be the better and richer if these techniques and wisdom can be transmitted to the discipline in general and to students in particular. Distillation of this wisdom into book form via the medium of real-world nontrivial examples is an obvious approach. This book is an excellent example of this genre.

B.P. Molinari
Australian National University

Book Reviews


This book is a translation of a 1983 French text. It presents a brief summary of formal approaches to the proof, construction, and analysis of programs. As stated in the introduction, ‘it is addressed to those who already have some experience in programming and who, wishing to pursue the subject at a more advanced level, are interested in particular in the proof and analysis of programs’. The emphasis is, correctly, on the construction rather than the proof of programs.

The first three chapters are concerned with iterative programs. Chapter One describes Hoare’s method (Hoare, 1969) for specifying and proving programs correct, applies the method to a simple example, and describes how to count the number of basic operations performed in a program. Chapter Two shows how to apply Hoare’s method to the construction of correct programs. Chapter Three extends the method to programs with arrays and procedures. The last three chapters are concerned with recursive programs. Chapters Four and Five deal respectively with the proof and construction of recursive programs. Chapter Six describes several methods for eliminating recursion.

Generally, the style is terse, formal, and clear. Some knowledge of logic is assumed. The presentation adopted is to give the formal definitions or statements of method first and then to apply it to one or more examples. The first application of a method to an example is done in complete detail; subsequent applications or extensions are done less formally. The examples used are familiar ones: three-coloured flag, integer square root, quicksort, Towers of Hanoi, Schorr-Waite algorithm, etc. The programming language used is essentially Pascal.

The two sections of the book invite comparison with two standard texts that address the same topic: Gries (1981) and Rohl (1984). The main difference in each case is that the current text is so much briefer than either of the others. Despite this brevity, it does summarize well much of the material in Gries. The main omissions are any guidance as to the construction of loop invariants and the more complex examples in Gries. On the other hand, more attention is given to the analysis of programs. Less important differences are the use of Pascal rather than Dijkstra’s guarded command language and the use of Hoare’s E[P]S notation rather than (Dijkstra’s) weakest preconditions, though these are briefly described. The main differences from Rohl are the more formal treatment of program proof and construction, and the smaller number of programming and transformation paradigms described.

In summary, the book presents a good brief account of its subject matter. As with many other books on the topic, the absence of complex examples leaves the reader to speculate as to the ultimate value of the formal approaches described. The style tends to be narrative, making it slightly inconvenient as a reference. Similarly, there are no solutions to the exercises, making it inconvenient as a text. There are indices to the notations and examples used, but not a general index. There is a short bibliography that includes the standard works in English and several works in French. Overall, the book should be a useful reference for courses using formal methods to construct and analyse programs, or for anyone wanting a brief introduction to this topic.

References


This book is an edited version of the Proceedings of the Sixth European Conference on Artificial Intelligence held in Pisa in September 1984. It consists of selections from the prize-winning papers and the long papers in the Proceedings volume and is organised into the five main general areas of Expert Systems, Robotics and Vision, Cognitive Modelling and Learning, Natural Language and Knowledge Representation.

The main strength of the book is the diversity of both problems addressed and solutions proposed to solve them. A random selection of problems are:

1. Modifying a pre-established schedule in order to minimise the deterioration of train traffic.
2. Knowing how to test for and regulate drilling fluid properties that influence characteristics of bore holes.
3. Extracting a tightly fitting peg from a hole.
4. Drawing a scene on a screen from a natural language description.
5. Construction of structured theories that can provide explanations of phenomena in terms of abstract entities and concepts.
7. The role played by examples in reasoning in various fields including mathematics, law, linguistics, and computer science.
8. Solving the Three Wise Men Puzzle.

The main strength of the book is also its primary weakness as in general the breadth of the material far exceeds the depth of explanation. I found some of the papers dissatisfying due to their brevity, while others could only be described as progress reports. However, due to this diversity there is bound to be something of interest to general computer scientists, or to AI specialists wishing to broaden their understanding of other sub-fields in AI.

P. C. Brebner

University of New South Wales


Cox is concerned about the lack of software reusability and provides a well-reasoned argument why encapsulation and inheritance (the principles of object-oriented programming) can have a positive impact on software productivity. In defining object-oriented programming, Cox discusses Smalltalk-80, ADA and C++. Although Simula was the first programming language to use the object metaphor, the Smalltalk family of languages developed at Xerox PARC introduced the term object-oriented language and has been the major influence on subsequent developments. Cox explains why ADA is not object-oriented even though it does provide some support for data abstraction through the package construct.

The book serves primarily as an introduction to Objective-C and its intended style of usage. Objective-C is a commercial precompiler that grafts Smalltalk-80 encapsulation, inheritance, classes, messages and objects onto the C programming language. The technique is claimed to be applicable to other conventional languages. After the first three introductory chapters, the object-oriented concepts are presented in Objective-C terms. A large library of class definitions is provided with Objective-C and some of these are explained in chapters seven and eight.

Cox argues that object-oriented programming is an attractive approach for interactive user interfaces, especially those based on icons. Indeed a primary objective of the Smalltalk research was development of programming tools and environments for designing user interfaces. Chapter nine explores how the object-oriented methodology assists the designer. Considerable benefit is obtained by separating the application’s functionality from the presentation level. This allows different styles of presentation to be provided, and perhaps more importantly, for a standardised user-interface to be acquired from a library. Through the object paradigm, the standard user-interface can be customised where it proves desirable but consistency across applications becomes the rule rather than the exception. Apple Computer have taken this approach with the Macintosh, the software library MacApp and the programming language Object Pascal. More details can be found in the book by Schmucker (1986). Objective-C will be available on the Macintosh.

I found this book easy and enjoyable reading in the early chapters with the concepts of object-oriented programming well motivated. The later chapters contain considerable detail that is important for a user of Objective-C but that may overwhelm the casual reader. An alternative introduction to the concepts and style of object-oriented programming in Objective-C can be found in articles by Cox (1984, 1985).

References


The rate at which computers have become involved in the development of manufactured goods at all stages from conceptualisation to delivery schedules has been rapid, widespread and dramatic. Already a multiplicity of specialist areas in the field of CAD/CAM (or Computer Integrated Manufacture to use a more general term) has arisen. There has been a need for a book such as this to provide an introduction to the most important areas in a reasonable number of pages. Mr Haigh has set out to provide that introduction and extend the descriptions to a moderate level of technical sophistication.

The early part of the book deals with the computerisation of drafting. This is covered from the fundamentals of software components of screen editing and scaling, to a description of the different forms which can be adopted for 3-D solid modelling. This section of the text is well supported by 60 figures showing examples of the equipment currently in use and stages in the preparation of drawings and illustrations.

The place of computerised techniques in the design process is illustrated using finite element analysis as an example. The formation of a simple finite element mesh is shown and the discussion is then extended to show its application to a problem involving the stress distribution in a curved cantilever. A further chapter shows how machine tool control programmes are developed (again supported by numerous practical examples) and outlines their integration with the computerised drafting techniques discussed earlier in the book. A brief survey of robotics is also included. Scheduling, materials replacement, process planning and quality control are referred to in passing. These areas may well be regarded as the all important bonus for undertaking computerisation in the first place. In all fairness it must, however, be accepted that a book of this size cannot cover every area of CAD/CAM in detail.

I believe that this text will make a good introductory text for those Technical College and University Students who will later be involved in manufacturing. Whilst not setting out to be an exhaustive treatise, it does lay down a balanced framework on which later courses can be built. Those people who are already involved in the computerisation of manufacturing will probably wish to proceed directly to the specialist papers in professional journals and specialised texts covering their own specific problems. The strength of this book is its smooth transition from basics to a moderate level of understanding.

Jacob A. Carmel
University of New South Wales


New problem-oriented disciplines tend to develop in fragmentary ways with each problem treated on an ad-hoc basis. A mathematician wandering around at this stage may be more of a hindrance than a help. After the initial burst of activity however, it is important that the discipline become integrated and placed in a more general context. Mathematics is the appropriate language for this integrated description.

The title "Computer Aided Geometric Design" is applied to the more mathematical aspects of Computer Aided Design, a problem-oriented discipline focusing on the automation of design processes. Surfaces in CAGD '84 contains the proceedings of a symposium held at the Mathematics Institute, Oberwolfach, FR Germany. The papers are concerned with the design of surfaces and the representation of surfaces. Design involves making interactive changes to surfaces which are displayed in real-time. Representation involves viewing surfaces in order to understand them and to pick out important features. The strategy used for the construction of surfaces is either interpolation or approximation. Interpolation requires surfaces to match the data exactly whereas approximation implies some smoothing of the data values.

The first paper by R.E. Barnhill gives an excellent survey together with some new results and open research questions. He emphasises the interaction in this discipline between quantitative data (such as scientific measurements) and qualitative data (a 'good' shape). The history of surface representation by a network of patches is traced from Coons and Bezier to the present.

The remaining papers in the volume are grouped according to content and emphasis. The first set considers approximations defined over triangles using Bernstein-Bezier representations. This set is the most mathematical.

The next set of papers is concerned with bivariate B-splines and general bivariate surfaces and is more algorithmic in nature. It is illustrated with 'good' diagrams. The emphasis here is on algorithms for interactive graphical display of smooth surfaces representing scattered data. The techniques range from successive smoothing of a piecewise constant surface to detecting undesired domains on a surface using orthotomic curves (well known in optics apparently).

Six short papers on tensor product surfaces cover implementation issues including the development of an international standard for a CAD system data-exchange interface. This standard is desirable to simplify the exchange of curve and surface representations between systems.

Most of the final set of papers are of a practical nature including several specific applications. They address problems such as surface-surface intersections and the determination of visible segments of regions.

This volume should prove to be a valuable reference for those interested in Computer Aided Geometric Design. Many of the papers include a well-written introduction together with a good bibliography which provide entry to the area even for those somewhat unfamiliar with the material. The diagrams are clear and informative including several in full colour.

Glenn Johnson
Macquarie University


This book is interesting in concept and somewhat philosophical in presentation. As suggested by its title it seeks to relate the computational needs of digital image processing to developments in hardware architectures, and the potential for VLSI implementation of those architectures. Clearly this is an important symbiosis and one that will continue to be increasingly so with the expansion of the data volume in typical images. In the past five years we have witnessed in space imaging for example a tenfold increase in the data size of Landsat scenes of the earth's surface such that a single image now represents 230 Mbyte of data. With current developments in so-called imaging spectrometry and the proliferation of imaging instruments expected in the next decade associated with space station and related initiatives, the need for computational efficiency will be of major importance, as it will also be in the fields of medical imaging and machine vision. Storage is not a consideration so much as a means by which image data may be manipulated and analysed on a time scale suited to applications. As is the theme of the book, this will drive architectural considerations and, given the simplicity of the basic operations into which most image processing tasks can be decomposed, will lead to means by which concurrency can be exploited; VLSI structures are well suited to that role.

Organisation of the book is good notwithstanding there being different authors for each chapter. The first and last chapters are entitled respectively Whithers Image Processing and Whither Image Processing, placing the current status on the field and its future trends in perspective. Chapters 2 to 4 essentially form the body of the treatment. The first of these is an overview of image processing algorithms, the second covers VLSI architecture and the third discusses the transition or mapping from algorithms to architectural structures. These are well treated in relation to the nature of the book. Then follow two applications chapters - one on computer vision and the other dealing with correlation of synthetic aperture radar (SAR) raw signal data into image format. The former is a natural choice by which to illustrate the book's theme, while the latter
might be regarded as a little idiosyncratic since the nature of the processing operations involved in SAR correlation is different from that of image processing for restoration or analysis purposes. Nevertheless it is an interesting treatment and does concentrate on hardware issues, in contrast to many existing treatments of SAR processing. The chapter on computer vision is important in principle; however as might be expected it is based heavily on algorithms and does not contain the balance of treatment on architecture that might have been expected in a book with this title.

A minor omission from the treatment is a mention of current commercially available image processing pipeline architectures associated with display systems. Composed of just function memories and hardware adders these allow many of the traditional image processing operations to be carried out in near real time. In the context of the book these would have been of historical interest. Nevertheless this is not a deficiency; the book is well put together and should be of particular interest to the student of hardware based image processing.

J.A. Richards
University of New South Wales

DATE, C.J. (1986 (fourth edition)): An Introduction to Database Systems, Volume 1, Addison-Wesley (price unknown).

The first edition of Date's Introduction appeared in the mid-70's. For its time it was an excellent book, and it was often recommended as a good introductory overview. Subsequently editions two and three appeared. The former preserved the character of the original, but the latter devoted much of its space to System R and related topics.

In the mid-80's, the third edition neither acquainted the user with the theoretical issues of the day, nor did it adequately cover what may be encountered with commercial offerings. Editions two and three had corrected and elaborated, but they had not reversed.

Edition four is quite different; it is virtually a new work. It does introduce some modern issues (distributed databases, optimisers, view updating, concurrency), and it is not preoccupied with a particular research endeavour. There are many balanced comparisons between different approaches. For example, the different paths taken by the developers of DB2 and Ingres are compared with the strengths of each noted.

The book has a relational slant. Date outlines in a relational context many concepts that have a broad application. The non-relational systems are given 101 out of the 623 pages. There is a chapter on each of: inverted files systems, hierarchic systems and network systems. The treatment here is regrettably brief.

In 1982, Date published a second volume of his Introduction. This was a series of essays that would interest the database specialist. Some of the notions from that volume have now found their way into Volume One. It is to be hoped that the author will soon revise the second volume, as the serious student of databases needs both volumes. Though they can usefully be read separately, they go well as a pair.

Date's style of writing is simple and he does not embroider his paragraphs with jargon. This has much to commend it, but this approach also does not leave the reader prepared for the style found in the literature. There will be readers who will lose by not being asked to think in sets, or who would have gained by pondering the difference between intensions and extensions. Likewise, the practice of 'definition by illustration' is not appropriate for a subject that has its basis in the precision of set theory.

For some years now, I have used Volume One as a textbook. The choice was made initially with much misgiving, and it was largely dictated by an affordable price. I can now recommend the fourth edition with no misgivings.

John Hiller
University of New South Wales


This is primarily a collection of eighteen papers and edited transcripts from the April, 1984 (London) seminar of the same name, augmented by six solicited contributions from well-known academics, who did not attend. The addition of these papers appears to have been motivated by a desire to add some quality (and hence appeal) to a book which would otherwise be left a bit short on substance.

As the editor tells us, the seminar was oriented towards industry, not academia, and attempted to cover 'recent developments' in problem-solving methodologies, resulting in a book which is presented in three parts: Expert Systems, Decision Analysis and Mathematical Programming. Recency is debatable, as the proceedings have taken over two years to appear, even without a formal refereeing cycle.

The emphasis is clearly on the expert systems approach (184 pp./16 papers), assisted by the solicited papers — all in this category — which were received at least 6 months after the seminar. The papers in this section from the seminar proper are mainly brief descriptions of commercial products by their vendors, such as PROSPECTOR (SRI), REVEAL (Tymshare/ICL), EXPERT-EASE (ITL/Export Software Intl), and SAGE (Systems Designers/ICL). Other papers give (by now) dated information on the aims of the Alvey Programme and the Japanese Fifth Generation Project.

The best illustration of how this book has become dated before publication is in Peter Hammond's solicited paper, 'Representation of DHSS Regulations as a Logic Program', which is basically a reprint of a paper which previously seen service in the 1983 BCS Expert Systems Conference, and before that, in a 1982 IC London technical report. Those who are interested in the application of expert systems techniques (and Prolog) to the interpretation of legislation would do better to refer to the more recent paper by Sergot et al. (1986) to which Hammond contributed.

On a more positive note, the main contribution of the seminar has been to create an awareness that not all problems are best attacked using one's favourite methodology. In my own work, I have noticed that progress has been hindered by this lack of communication between expert systems researchers and those in other disciplines.

A book for libraries with unlimited budgets.

References


John O'Neill
Australian Defence Force Academy


This book is something of a rarity — a book written by academics about the structure of a particular operating system and how to exploit it. It differs from the more common "hackers' handbooks" in that the technical information is presented in a more complete and systematic way, illustrated by well documented examples. It is unfortunate that such books are not more common.

This is not a book for beginners. It assumes some familiarity with Z80 assembly language programming, and with CP/M and its terminology. These assumptions should provide no difficulties for most CP/M users.

The book should give such people a deep understanding of how their operating system works.

Considering the small size of the book, a vast amount of technical detail is presented, and the information is illustrated by copious examples. After a very brief description of the Z80 processor, the authors describe at greater length the internal structure of CP/M. BDOS (basic disc operating system) and the subroutines of the standard BDOS library are described in considerable detail, since an outstanding of these subroutines is required in order to write input-output programs that are compatible with CP/M. A chapter containing six complete examples of such programs follows. For readers who wish to delve deeper into the operations of CP/M, the book concludes with a chapter on how a typical CBIOX (customised basic input-output system) is structured. This is a long chapter which contains many examples of subroutines to perform basic control functions for the common input-output devices (console, printer, disc) and describes bootstrap programs. The chapter also includes detailed information about debugging and installing a new CBIOX and formatting discs. It concludes with some examples of useful additions.
that users could incorporate into their own CBIOS. Two short appendices
give the Z80 instruction set and the ASCII character set.

The book contains a great many listings of Z80 assembly language
programs for use under CP/M. Many of these would be immediately
useful for most CP/M users (for example, a program that restores erased
files if their disc space has not been reused). Others would appeal to more
specialised users (for example, a modification to the CBIOS that enables
multiple disc formats to be recognised). All the programs illustrate inter­
esting points about how the CP/M system works and how users’ pro­
grams can interface with it.

This book is just the thing for CP/M users who wish to learn some­
ting about how their operating system works and how they can exploit it
to suit their individual requirements.

Bill Beaumont
University of Adelaide

Press, 241 pp., $5.150 (hardback).

A recent unsatisfactory ARG5 interview having convinced this reviewer
that he should confine his acts of judgement and review to areas in which
he could claim some degree of expertise, I accepted the request by the
editor of this Journal to review the above title with misgivings. With what
authority could I pass comment on a presentation in combination of the
Ada™ language and concurrent procedural programming methods, being
a somewhat monocural advocate of functional programming and as
ravid an Ada-basher as any other computer scientist with modish procliv­
ties? Fortunately, I was about to commence a project which required
knowledge of both those areas, and so finding myself a member of the
very target audience for this book (software development professionals
of one form or another), realised that I was specially placed to assess its
worth. Now follows my assessment, presented in all due humility.

The title belongs to The Ada Companion Series from CUP. Its compan­
ions, identified in the preliminary pages, include conference proceedings,
bibliographies and monographs such as this. They are announced with
the claim that ‘There are currently no better candidates for a co-ordinated
low-risk and synergetic approach to software development than the Ada
programming language.’ If you now share my initial fears that a collec­
tion of pro-Ada propaganda tracts was to follow, this book puts them to
rest. (The one other in the series that I have seen is similarly reassuring.)
Ashton-Tate have produced a number of books to support the Frame­
work II. Often a little help is needed to bridge the gap between the tutorials supplied on
disk and the manuals. This is such a book. It provides useful hints and a
good guide for users who want to get into action rather than spend a lot of
time reading manuals.

The style of the instruction is clear, except for a few typographical
errors, and the reader is led very quickly into the basic features of the
language. The book is in three parts:

1. Introduction to the Framework II environment and commands, covering
Text frames, Outline frames, Databases, Spreadsheets and
Graphs. This section will get the new user familiar and comfortable.

2. Common Wordprocessing tasks from reports to books. This section
shows how wordprocessing, and in particular Framework II, can
enhance their capabilities if they wish. A complete application is
available “add-ons”, and the use of the FRED programming lan­
go to automate many of the tasks. This section will allow users to
enhance their capabilities if they wish. A complete application is
supplied for users to type in, and may be available on disk as was the
case for another book, to demonstrate the full power of the FRED
language.

Overall, Framework II for Writers is a helpful book for new users of
Framework II and can show some more experienced users some tricks (I
found five good ideas). I would recommend it to all new users, and could
be useful to other users who, like me, like to refresh their memory on the
wide range of features of the Framework II package.

land, 281 pp., $US37.00.

Office automation is a broad term used to describe the application of a
number of computer-related technologies to improve the effectiveness of
office workers. One of the technologies is computer networking which
allows a network of computers to convey messages at relativistic speeds
around a network of workers, who are often separated by large geogra­
phic distances. Network design for office situations is a real engineering
task involving difficult tradeoffs between cost and capacity. It is difficult
and challenging, and if this is your problem, then you are almost certainly
looking for all the help you can get, and this book may just catch your
attention.

Forget it! Its title is one of the most misleading that I have encountered
for a long time. The content is the proceedings of a symposium held in
Softi, Bulgaria, in September 1984. The papers, as usual with collections
from this publisher, are printed from copy supplied by the authors. I am
personally suspicious of office automation papers that are prepared using
a manual typewriter and are full of obscure mathematics. As befitted the
location, most of the speakers came from the Eastern bloc — only seven
out of 32 papers came from Western countries (see each from Portugal,
Japan, USA, West Germany and the Netherlands, and two from the UK).

Students of Eastern bloc computing technology may be fascinated by this
volume but it does not cast much light on contemporary Western prac­
tice. It contains very little about office automation — and it seems to be
mainly about networks that the authors are still thinking about. I found
very little in it to interest me, and I think you can quite safely ignore it also.

John Lions
University of New South Wales

WILLIAMS, FREDERICK (1986): Framework II for Writers, Ashton­
Tate Publishing Group, California, 220 pp., $36.00 (paperback).

Ashton-Tate have produced a number of books to support the Frame­
work II package, but this is the first I have seen for the Framework II. Often
a little help is needed to bridge the gap between the tutorials supplied on
disk and the manuals. This is such a book. It provides useful hints and a
good guide for users who want to get into action rather than spend a lot of
time reading manuals.

The style of the instruction is clear, except for a few typographical
errors, and the reader is led very quickly into the basic features of the
package. Building on this foundation Dr Williams then shows how
Framework II can be used for most writing tasks, and how some of the
advanced features can save time and produce a better product.

The book is in three parts:

1. Introduction to the Framework II environment and commands, covering
Text frames, Outline frames, Databases, Spreadsheets and
Graphs. This section will get the new user familiar and comfortable.

2. Common Wordprocessing tasks from reports to books. This section
shows how wordprocessing, and in particular Framework II, can
enhance their capabilities if they wish. A complete application is
available “add-ons”, and the use of the FRED programming lan­
go to automate many of the tasks. This section will allow users to
enhance their capabilities if they wish. A complete application is
supplied for users to type in, and may be available on disk as was the
case for another book, to demonstrate the full power of the FRED
language.

Overall, Framework II for Writers is a helpful book for new users of
Framework II and can show some more experienced users some tricks (I
found five good ideas). I would recommend it to all new users, and could
be useful to other users who, like me, like to refresh their memory on the
wide range of features of the Framework II package.

Alex Cruickshank
South Australian Institute of Technology

References
Ada Tasking, Technical Report 63, Department of Computer Science,
University of Queensland.

Paul A. Bailes
University of Queensland
Letters to the Editor

LETTER FROM DR KIDMAN
I regret that I have had to inform the editorial committee of the Australian Computer Journal that I am unable to take up the position of editor on 1 January 1987.

I had agreed to do this job on a voluntary basis after retirement as a service both to Australian academics in Computer Science and to members of the Australian Computer Society, of which I have always been a member. However, I have recently become aware of circumstances which make it impossible for me to undertake the editorship. I greatly regret the inconvenience caused.

Idealistically I have seen our journal as akin in aims to the British Computer Society’s Journal and the American ACM’s Communications, but of course these must be long term aims for the Australian Computer Journal, particularly with regard to content and circulation; editorially the ACJ is an excellent publication by international standards. Improved international recognition will only come from full support by all Australian computer scientists. The Australian Computer Society, like its companion societies in Britain and America is to be congratulated on its foresight in publishing our journal, which I hope continues to progress; I wish it a very successful future.

Barbara P. Kidman
Department of Computer Science
University of Adelaide, SA

MORE ON EDITORIAL POLICIES
I note with interest and support the indication in the Guest Editors’ Introduction to the ACS, November 1986, that the ACJ will create a department for publication of Information Systems papers.

However, it is not, as the editors state, a problem of perception that the ACJ is currently a computer science journal. Rather, it is inherent in the editorial structure and philosophy. As evidence I would point to:

1. Under types of material, the word ‘computing’ appears five times. Other possible key words, like information system, telecommunications, office automation, and business systems, do not appear at all.
2. The editorial board appears to have only academic members.
3. There has never been, to my knowledge, any indication of the background of reviewers. What proportion represent the views and interests of industry?
4. In the May 1986 issue, the editor stated ‘writing serious articles for the permanent record is an exacting task’. This is no doubt true — but it gives non-academics, unpractised in the art, no opportunities to obtain experience in this activity.
5. The audience for the papers published does not appear to align with employment patterns in the industry. I analysed four years of papers, from May 1981 to February 1985 (see my paper in the First Pan Pacific Conference Proceedings, p. 1339). In this I found that 45% of the ACJ papers addressed technical systems builders (data management, communications, operating system specialists) whereas these people represent only 3.8% of the computer related staff numbers, or 11.4% if you ignore data entry/WP/terminal operators. (See ‘Data Processing in Australia’, p. 25. R. Kriegler, R. Blandy, P. McGavin, J. Ryan, pub. Allen & Unwin, 1986). The same sized group of people, DP Managers, received only 7% of the ACJ’s attention.

I do not want to be negative — I applaud the changes. However, I would like to see the ACJ take a strong position of support, and look at overseas examples, particularly the MIS Quarterly and the Communications of the ACM, where differentiations are made in the reviewing and publication practices between papers of interest to computer specialists and to IS practitioners.

I wish you well and would be happy to help in the transition.

Michael Sager, MACS
Management Information Systems
Deakin University, Victoria, 3217

MORE DUMB, DEFENCELESS CREATURES
I was fascinated by the correspondence published in the November 1986 issue. I do not, of course, know what Mr Jones said in his first letter but I can agree that he went overboard in his second.

Academics do have a right to publish and the medium of the Journal is provided for anyone wishing to exercise that right provided that appropriate standards are maintained. The Journal and the ACS has the right to expect this.

However, Professor Lions also betrays an academic frustration or intolerance with those of us who cannot or will not publish material. As an independent consultant in geophysics and therefore merely an advanced user of computers I just cannot afford to publish any new or unique developments that I have made either in my own specialty or in computer applications of them. Commercial reality, not to mention any confidentiality which may attach to the examples, forbids it and I resent being called a leech as a result.

Those academics who take the ‘they’re all leeches’ view need to consider just who is arrogant and unrealistic. After all they do have their freedom to publish, a medium and readers. So share it.

I can provide an example of unreal expectations. I recently submitted a review and experience-advice paper to another journal which was directed at applications and misconceptions. It was wholly constructive and drew on long experience. The academic referees accepted the
Letters to the Editor

I am led to believe that recent medical research in the UK suggests that leeches may provide some very useful evidence in the fight to combat heart disease. I must protest, therefore, that your response to the letter from Mr Brian Jones (ACJ Vol. 18, No. 4, November 1986) was most unfair to leeches. I do hope you will be gentlemanly enough to apologise to those small creatures.

Mr Jones, of course, was equally unfair to moles. After all, perhaps they keep to the dark BECAUSE they can see the real life out in the World!!

The two animals mentioned in the dialogue between yourself and Mr Jones do bear further study. Like most of 'lesser' inhabitants of this Earth, they do seem to have a very good idea of their roles in the scheme of things, and of what they need to do to co-exist and to survive. They do seem to appreciate that their environments are systems of interlocking, inter-dependent parts; that few things in their ecosystems function very well in isolation from anything else.

What a contrast computer 'professionals' make! On the one hand, we have the spectacle of some non-academics (for want of a better term) bemoaning the publication of the ACJ, and claiming that commercial research will not be published before it reaches obsolescence because of competitive reasons — a line of argument that should cause some bemusement. I am sure, to the editors of the IBM Systems Journal, the Proceedings of the IEEE, etc. On the other, we have an 'academic' suggesting that those who don't like the ACJ should leave the ACS — a line of argument that can do little to enrich the ACS in the eyes of many in the non-academic world.

Surely one of the things Australia needs desperately at the moment is a far closer liaison and co-operation between industry and universities. We need this to enable us to compete in both domestic and foreign markets. If we can compete successfully, we can create wealth, jobs, homes, sustenance and the opportunities for learning and growth for ourselves and our children. If we cannot compete, we will surely lose all these things to others more willing to apply their efforts and their intelligence. In the USA, the UK, and Japan, (and many other countries) there is a very close liaison indeed between universities and industry. One role that the ACJ can fill is to provide a link between the work of universities and industry. If that task is not done very well at the moment (and I make no judgement on that), it can only be done by the improvement of the Journal, not its contraction.

Arguments about whether the Society needs the Journal more than the converse are really just a waste of time and reflect no credit on any section of our industry. Surely effort put into such arguments could better be directed to improving the ACJ. If so many of the serious technical papers published are difficult to understand, surely they might be criticised, analysed, and commented upon in the ACJ. Perhaps the addition of commentaries about papers, commentaries aimed at highlighting the principal points arising from a paper and suggesting the impact of those points on commercial work, could be a worthwhile innovation.

I hope there are better ideas than that for improving the ACJ, and the Journal should seek constant improvement. That has to be a much more productive course than any dialogue about its destruction.

Rob Charlton, MACS
Westleigh, NSW, 2120

LA TROBE UNIVERSITY
School of Mathematical and Information Sciences
LECTURER IN COMPUTER SCIENCE
(Tenurable)

The appointment will commence as early as possible from the beginning of 1987. Applicants should have a PhD in Computer Science or related discipline and demonstrated research ability. Duties will include both undergraduate and graduate teaching and supervision of postgraduate students.

Applications will be considered from suitably qualified persons with a background in any area of computer science. The department is developing centres of excellence in Real Time and Co-operating Computer Systems and Artificial Intelligence (especially IKBS and Computer Vision) within a lively research environment.

Further information may be obtained from Professor T. Dillon, Chairperson, Department of Computer Science, La Trobe University, Bundoora, Victoria, 3083, Australia.

Closing Date: 2 March 1987 (Reference No. A0/032/001).
Salary: $27,859-$36,600.
Applications (marked confidential) including reference number, names of three referees and Curriculum Vitae should be forwarded to the Staff Officer, La Trobe University, Bundoora, Victoria, 3083.

Equal opportunity is University policy.

COMPUTER SYSTEMS GIVE PIN POINT ACCURACY IN FLEET POSITIONS

A sophisticated network of computers, satellites, radio and facsimiles were all part of the technological support for the 1986 AWA Sydney Hobart Yacht Race.

Race sponsor Amalgamated Wireless (Australasia) Limited, Australia’s electronics and communications company, developed the network to ensure that the yachtsmen and women have the best possible support systems behind them as they take part in one of the world’s most prestigious long distance yachting events.

For the 1986 AWA Sydney Hobart, the centre of the support organisers system was again the radio relay vessel TV Wyuna, loaned to the Race by the Australian Maritime College and manned by an all-volunteer crew.

On board were four AWA radio operators and technicians to operate and service the equipment including a ship-earth satellite station, two 16-bit micro computers, a statistical multiplexer and an array of the latest AWA radio communications equipment.

Increased use of the satellite systems has revolutionised ship-to-shore communications and this is put to great advantage in positioning the yachts in the AWA Sydney Hobart.
AUSTRALIAN COMPUTER JOURNAL

Subscription/Change of Address Form

Name.................................................................
Current Address..................................................

[ ] Please enrol me as a subscriber for 1987. I enclose a cheque or bank draft for $425.00.
[ ] Please enrol me as a subscriber for 1987, and send my copies by airmail. I enclose a cheque or bank draft for $440.00.
[ ] Please record my new address as shown above. I attach below the mailing label for the last received issue.

ATTACH LABEL HERE

Send all correspondence regarding subscriptions to the Circulation Manager, PO Box 319, Darlinghurst, NSW, 2010, Australia. Photocopies of this form are acceptable.