*Information Technology: Research and Development* (1982), *1* (301–324)


# EXPERT SYSTEMS: AN EVOLUTION IN INFORMATION RETRIEVAL

T. R. ADDIS

*Man-Computer Studies Group, Brunel University, Uxbridge, Middlesex UB8 3PH, UK*

(*Received 21 April 1982, revised 2 August 1982*)

## ABSTRACT

This paper reviews the field of expert systems and concludes that the range of systems represents different degrees of enhancement to an information retrieval system. Extended relational analysis is briefly introduced in order to provide a method of comparison between an industrial database, a diagnostic aid and a medical consultation system. It is recommended that expert systems should be considered as a means of communicating group practices to trained users rather than simulating an expert to aid novices.

## 1. INTRODUCTION

It is claimed by many in artificial intelligence that the essence of intellectual ability can be captured within a computer program. To make such a claim requires that this ability can be shown, at least in principle, as a formally expressed rubric. A distinction may be made which makes clear the role of the rule in the representation of actions and activities. This distinction is between naturally occurring performance and its representation within a formal system. Naturally occurring performance such as the moon's orbit round the earth can be described within a formal system of mathematics and simulated within the computer. However, there is no suggestion that the moon 'calculates' its position with respect to time or that there are necessarily any physical representations for such calculations (Dreyfus, 1972). A similar distinction can be made between human behaviour and its simulation within a computer. Certain kinds of restricted language activity, for example, can be monitored and described but there is rarely any conclusive evidence that there are physiological or psychological correlates to these descriptions (Gough, 1966).

It is considered by some that human approaches to problem solving should obey the rules of logic and that any deviation from these rules is considered an aberration. However, much of human behaviour lies outside the domain of standard logic (*see* Toulmin, 1976; Johnson, 1982) and this includes the intellectual ability of an expert within a field of endeavour. It is this kind of expertise that knowledge engineers are trying to capture for expert systems in order to simulate the essence of experts' skills. An expert system can be viewed as a method of recording and displaying human competence where human competence is the *agreed* ideal behaviour within specified environments (Johnson, 1980; Hartley, 1982). An expert system does not exhibit natural human performance and consequently does not represent a replacement expert as has been suggested elsewhere (Duda *et al.*, 1980). Consequently, an expert system should be considered as a tool and a means of coherent communication of the standards of competence within a group of experts in a particular field. Further, since the computer has no means of providing natural performance (except as a computer) this transmission of standards of competence can only be sensibly interpreted and used by someone trained in the field. It is for this reason that no expert system has been produced for completely naive users. However, an expert system ought to go beyond offering mere standards of competence by also providing all the facilities of the computer. Just as a pilot requires a check list to ensure that procedures are followed so expert systems can help to prevent human error.

Information retrieval (IR), on the other hand, is an area concerned with providing users with a tool for extracting relevant information from a large pool of facts. The pool of facts is usually a collection of documents that are individually characterized by certain features (index terms). The central problem in IR is the specification of the user's requirements in terms of these features. The simple logical concatenation of features to identify the set of documents of interest to the user has proved not to be completely satisfactory in practice. The problem is caused by the nature of the features: the fact that they are words. Words in isolation can have many meanings and when combined into phrases they can be cajoled into many subtle variations that cannot be easily simulated by logical connectives. Consequently, the user's specifications to an IR system can often return much irrelevant material, and attempts at refining the original request can result in no response.

One possible view of information retrieval objectives is 'to create a model of the documentation set in such a way that it is homomorphic with respect to user requests'. This model needs to be more than just a list of index terms, but to be terms in relationships. The introduction of a thesaurus of index terms with relationships such as 'superset' and 'subset' suggests the development of data structures similar to semantic nets. These semantic net-like structures embody the essence of the subject area of the document set in a way that is sufficient for selecting (Oddy, 1977). The techniques for creating such structures bear resemblance to methods of knowledge elicitation now evolving within the domain of expert systems (Sparck Jones, 1973). In this area of elicitation IR might be able to offer useful experience to those concerned with expert systems particularly since there seems to be an evolutional convergence of what may seem to be two distinct sciences.

## 2. THE RANGE OF EXPERT SYSTEMS

One of the first practical expert systems was called DENDRAL (Buchanan and Feigenbaum, 1978). The work on DENDRAL started in 1965 at Stanford and its task was to enumerate plausible structures (atom-bond graphs) for organic molecules given information from analytic instruments and user-supplied constraints. The many possible structures that have to be explored required using certain graph theoretic notions in order to exhaustively consider all possible solutions. The domain of search was not an explicit set of known organic structures but an implicit set derived from a theory of how such structures are formed. This approach was formal and strictly non-human since it represented an intellectual ideal of how to achieve a solution which was beyond human capability. The excellent performance of DENDRAL was achieved by a symbiotic relationship between the competence represented within the system and the natural performance of the operator in guiding the system through its search task.

There have since been many other expert systems as offspring of DENDRAL. In the drive to create expert systems to model human thought processes the medical consultant MYCIN (Shortliffe, 1976) and the geological expert PROSPECTOR (Duda *et al.*, 1976) were specifically designed to represent and explain all their reasonings in human terms. In these cases the principle of generating vast numbers of solutions from some theoretical model was inappropriate. This was also done because it was assumed that *no* physician would (or should) consider using a proposed therapy and *no* geologist would recommend expensive mining operations unless they could justify the conclusions of the relevant expert system. The generation of an answer in a way that cannot be followed by the user was considered irresponsible. This pressure to simulate human recognizable reasoning produced several other systems. Many of these systems employed EMYCIN (empty MYCIN) which captured the framework of a general expert system. It just required the addition of knowledge.

Concurrent with these latest variations on the Stanford theme were systems developed directly from pattern recognition and IR techniques. The ICL CRIB system (Addis, 1980) using the new IR hardware CAFS was a diagnostic aid developed for engineers. The aid was able to help diagnose faults by retaining a vast accumulation of symptom patterns gleaned from many sources. Thus no model of selection was needed and justification could be supported by case law. Certain simple heuristics were employed to provide a conversation between the engineer and the system in order to guide the engineer towards a potential stored fault that fitted the observed symptoms. It is the use of these simple heuristics to provide feedback which differentiates this kind of diagnostic aid from a simple information retrieval system. The feedback is not just the presentation of the possible set of solutions but is also some guidance on how best to distinguish them and further refine the selection specification. At this level of interaction the appearance of the CRIB system was similar to that of the Stanford expert systems. However, the nature of the knowledge retained and the capability of each system were subtly different. CRIB was biased more towards the accumulation and use of factual knowledge as opposed to developing a rule-based competence model of a human skill (cf., Feigenbaum, 1980).

## 3. THE HUMAN FACTOR

The principal feature of expert systems is that they deal with problems whose specifications are complex, so that non-human methods can often be used to advantage. Complex problems usually lead to vast numbers of possible solutions, each of which must be examined with care—an assessment task that is often well beyond normal human capability. For example, DENDRAL is concerned with determining the molecular structure of organic compounds from the results of a mass spectrometer, where the problem is that the molecule gets broken down into untidy lumps (molecule fragments) which are recorded by the instrument. Consequently the many possible structures that have to be explored require formal searching techniques based upon the theory of graphs in order to exhaustively consider all possible solutions. Each solution is examined to see if it is a possible candidate for the particular spectral response of the molecule under test. Another example of non-human problem solving is MACSYMA (Bennet *et al.*, 1979), a program for evaluating integrals in mathematics. MACSYMA uses the Risch algorithm which is never employed by mathematicians because of its complexity. However, the Risch algorithm is considered the ideal approach for evaluating integrals and consequently reflects a standard of human competence. The competence described here is the formal method that generates solutions and is a model of the total set to be explored.

The medical consultant MYCIN (Shortliffe, 1976) and geological expert PROSPECTOR (Duda *et al.*, 1976) on the other hand were specifically designed to represent and explain all their reasonings in human terms. The requirement was to provide a model of solution generation that was dependent upon a balanced mixture of facts and rules for generating potential solutions (Michie, 1979). This was achieved by gleaning understandable rules of diagnosis from the human experts and providing a control structure which can use these rules directly. This technique was also used in most part by both DENDRAL and MACSYMA (see above). The control structure would not only draw inferences from these rules but could calculate some measure of certainty concerning the final conclusion. The significant principle behind this measure of certainty is that it was not based upon traditional probability theory, which relies on gathering statistical evidence but was intended to model an expert's 'belief' or 'disbelief' about a particular hypothesis given the accumulating evidence.

The control structure that handles this evidence, although more understandable to the user, does not in fact represent the manner in which human experts reason. The systems INTERNIST (an expert system for internal medicine; Pople, 1977), PSYCO (diagnosis of dyspepsia; Fox *et al.*, 1980), and PIP (concerned with oedemas and renal disease; Pauker *et al.*, 1976) have been developed specifically to mimic the human diagnostic procedure.

The main characteristics of all these systems (MYCIN, PROSPECTOR, INTERNIST, PSYCO and PIP) are goal-directed reasoning, sensitivity to patterns of symptoms and the abandonment of probabilistic (i.e., statistical) reasoning. MYCIN and PROSPECTOR will reason backwards from the goal (to determine the bacteria or ore deposit) to sub-goals and then sub-sub-goals in a way that is distinctly non-human. It is a form of reasoning through justification. PSYCO, on the other hand, attempts to reason in a forward direction in a way more akin to observed clinical behaviour. The importance of goal directed reasoning in either direction is that the user perceives a coherence in the questions being asked. The danger of the rule-based systems is that each answer to a question widens the circle of potential hypotheses that have to be investigated by the system with the consequence that the system can ramble on long after a human expert would have come to a decision. From an IR point of view it is a technique that would synthesize document descriptions (solutions) in a way that could be followed by the end-user and allow a controlled form of browsing.

## 4. THE COLLECTION OF KNOWLEDGE

All the systems mentioned above depend upon gathering knowledge in the form of rules (the 'production' rule technique). The advantages of these rules are that the knowledge concerning the particular area of expertise (competence) can be expanded incrementally and is in a form easily expressible by an expert. Associated with MYCIN is the program TEIRESIAS (Davis and Buchanan, 1977), which collects new rules from the expert, checks the consistency of the rules and follows chains of reasoning under the control of the expert in order to discover inadequate or inappropriate rules. An equivalent program, Meta-DENDRAL, is linked with DENDRAL. TEIRESIAS is quite subtle and intricate in its operation and this is achieved through a set of meta-rules used for collecting the diagnostic rules for MYCIN. These meta-rules are concerned with the general outline or form of the diagnostic rules and, in particular, which kind of diagnostic rules are associated together. They represent a generalization of the diagnostic rules which can be used to guide the MYCIN search strategy (i.e., only examine rules of a certain type under special circumstances). This principle of controlling the tracing of inferences at an abstract level before plunging into detail has been principally employed by ABSTRIPS a general planner for a robot system at SRI (Sacerdoti, 1974).

The problem with rule collection is that it can be a long and tedious business. What can often be collected more easily are examples of symptoms that lead to a diagnosis; in the case of fault-finding in machines as opposed to the medical treatment of illnesses, these examples are often readily available as fault reports. However, such reports are frequently inadequate, incomplete, inconsistent and out of date, simply because the effort of maintaining them is labour intensive. The incentive for engineers (or doctors in the case of medical records) to give accurate and complete reports is lacking because the usefulness of such records in their raw state is low. However, provided a system that will act is an automated clerk which will harmonize and analyse these reports in a way that makes them useful then the quality of reporting should radically improve. A technique used in IR for evaluating the usefulness of terms for retrieval has reappeared in the form of automatic induction (Sparck Jones, 1974). Work on automatic induction has started at Stanford University with the objective of deducing a set of diagnostic rules from a database of examples that can be used by an expert system (Quinlan, 1979).

The ICL CRIB and RAFFLES (Addis, 1980), on the other hand, have been designed to work directly on a database of examples without the need to deduce a set of rules. In this case there is no attempt at generating a solution. CRIB uses a few simple prefixed meta-rules under the control of the user to 'home-in' on a matching 'fault'. The user can draw on his own experience of the situation to guide the system to search for faults with particular attributes. In order to save online computer power, RAFFLES precompiles the CRIB process by generating an optimum fault-finding guide. Thus the RAFFLES technique provides an optimum indexing of terms. This achieves the same objectives (i.e., finding a matching fault) but with minimum online computer processing. The weakness of the initial RAFFLES implementation is that because it is precompiled the user lies outside the diagnosis activity. The user is thus driven as a passive observer. This tendency is an unwelcome by-product of most expert systems since it does not fully exploit the user's local knowledge or his performance ability. The main stumbling block is always the problem of accepting and 'understanding' free form descriptions of symptoms from the user in some kind of English.

Both MYCIN and PROSPECTOR have been deliberately structured so that their representations of rules are embedded in an organization suitable for 'natural' language dialogue. MYCIN converts the English form of the rules into a set of standard relations with a fixed set of attributes. This allows direct manipulation of the 'meaning' of the rules by the inference-making control structure, makes allowance of the context by constructing a 'context tree' from the conversation and uses this context tree to modify the generation of English to the user. However, the English to internal representation is weak and is only used for new rule generation where checks can be made by regenerating the English from this internal representation.

PROSPECTOR employs semantic nets to represent the rules but does not have any form of 'natural' language input at the moment. Although the intention of the designers and the potential of the system is to have free form English conversations, the problem of diagnostics has absorbed most of the designers' efforts. This representation of rules in standard form is nevertheless important in that it is open to manipulation, transformation and merging, thereby providing a strength in the reasoning capability and subtlety in the interaction that is not present in any text-based system. To simply store text that is merely hooked into the system via key words or other associated attributes is to create a system which is wide open to misinterpretation by users. There is always the possibility that the text, which is meant to 'stand in for' the key words and the key words' effect on the control structure, will imply more or suggest something different to that which was intended.

## 5. THE PROBLEM DOMAINS

Statistics and pattern-recognition techniques as applied to medical records have proved to be unsatisfactory. However, diagnostics that involve objective tests have proved tractable to these techniques (de Dombal *et al.*, 1974). Initial trials to determine treatment within a confined area of diseases from a large database of examples have had some success (Nordyke *et al.*, 1971) but generally such an approach is of little use. The problem lies in the social aspect of medicine and medical treatment that reflects the continuous modification of a defined competence. Not only are medical records inaccurate but diseases themselves change from year to year. To compound this problem, the usefulness of disease categories is not permanent but is a function of changing treatments and styles (Edwards, 1972). Thus, any statistical approach that depends upon collecting material over a long period is doomed to fail because the many combinations of patient types, diseases and treatments demand vast quantities of examples.

MYCIN accepts that no single 'cure' for any bacterial infection can be recorded since combinations of bacteria may exist (and often do) within a patient who may have certain sensitivities to drugs. Thus, MYCIN is designed to first find out which bacteria exist then determine the patient's sensitivities. From the physical attributes of the patient (weight and age) MYCIN can calculate the dosage of a selected minimum set of drugs (which do not affect the patient) to combat the bacteria. Although MYCIN is concerned with only a few bacteria (about 100), the combination of possible bacteria, patient sensitivity and drug combination can be very large. What is more, it can also represent a combination that may never have occurred before which may be further complicated by the progress of the disease and its different manifestations after past therapies.

The CRIB and RAFFLES technique can only respond with a tried and tested 'cure' but the problem domain is different. Although machine faults may interact with the software to produce different symptom combinations for the same fault, this interaction is both discrete and limited. It is discrete in that every machine is equivalent and symptoms are more often than not precise. It is limited in the sense that most conditions can be reproduced exactly so as to retrace events that lead to failure. This can never be done in the diagnosis of diseases where every patient is unique, symptoms are vague and diseases change. Machine (or system software) faults will often accumulate into one or two thousand *different* faults before new designs or modifications alter the environment.

Further, old designs (mods) are often kept going long after the engineers (or software specialists) have moved on to different interests. A more rigid or structured approach can thus be used in these cases whereas the rule-based method may well require too much processing to be practical.

## 6. THE STRUCTURE OF EXPERT SYSTEMS

The relationship between the different expert systems can be illustrated in a flow diagram consisting of rules that generate solutions which are then assessed for presenting questions to be answered or solutions to be tried. The flow diagram (Fig. 1) indicates that solutions can often be captured directly from the expert in the form of records or fault reports; usually, however, expert systems require the expert to describe the primitive decision rules that can generate these solutions from circumstances. In one sense these rules represent an equation of diagnostic behaviour in the same way that equations are used to predict the behaviour of bodies in flight. These descriptions of rules normally require a knowledge engineer to act as interpreter.

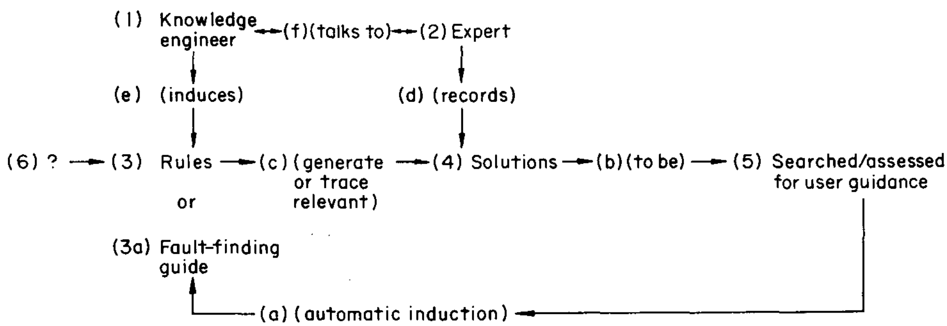


FIG. 1. The relationship between different expert systems

MYCIN and PROSPECTOR systems which include the update programs consist of most of these elements except (6), (3a) and (a). MYCIN retains the solutions recorded while the expert uses the system and although these are not used directly to provide therapy as with CRIB the records are used to check modifications to the set of rules. CRIB does not use rules to generate solutions whereas RAFFLES, CLS and INDUCE generate rules from a database of examples. An information system consists of the solutions only with no means of user guidance. A system that can generate its own rules, independent of examples or assessment routines, is indicated by the question mark (?). Such a system would require a theory of diagnostic behaviour that is good enough to produce precise equations. This theory is particularly required in dealing with new situations that lie outside the range laid down by the primitive rules or example solutions.

Thus, there seem to be three major categories of expert systems where each is defined by the class of facts retained. The first class—simple facts that are used directly as in CRIB (4,b,5)—can only provide answers that have been clearly recorded beforehand. The second class is simple fact interpolation that will use rules to apparently generate new simple facts that can be used (3,c,4,b,5), e.g., MYCIN and PROSPECTOR. It is fact interpolation because there is a well defined limit beyond which the simple rules cannot go: the rules cannot deduce new diseases or recommend startlingly new treatments. The third class of recorded experience is simple fact extrapolation (6,3,c,4,b,5) of which there are currently no practical examples, although the mathematical concept generator program AM claims to be a candidate (Lenat, 1977). This class would be concerned with generating new rules of discovery. Note that the closed loop (a) provided by automatic induction is used for reasons of efficiency rather than giving new insights. Thus RAFFLES, CLS and INDUCE are still within the second class of expert system.

An alternative method of describing these relationships between expert systems can show a finer dichotomy between the component parts. The CRIB system, for example, which includes the update program, also includes operations that form part of the MYCIN rule base. The alternative view, shown in Figure 2, considers expert systems as an evolution in information retrieval.

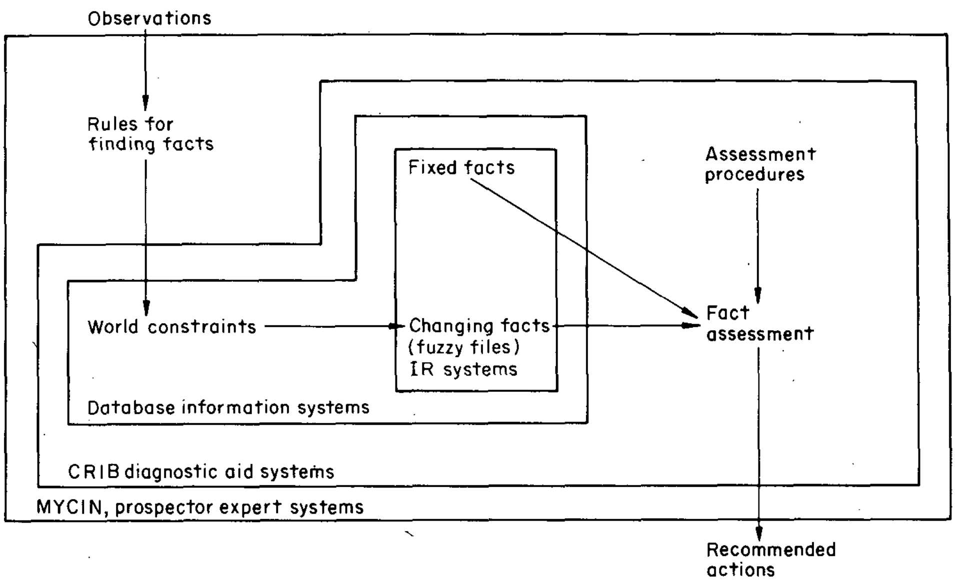


FIG. 2. Different types of retrieval systems

Each 'higher' type of retrieval system contains all the mechanisms of a 'lower' type. The simplest type of system, before entering into the categories of expert system, is a set of files which can be accessed by retrieval programs. These files can be 'fuzzy' files in that attributes may have several alternative values with some level of 'certainty'. Such systems that use non-fuzzy values represent the typical commercial use of computers for keeping and processing company data. The next type is the management information system and the important addition is a protective layer of constraints around the files. It is these constraints (world constraints) that prevent the data in the files from being corrupted and make a set of files into a database. Such constraints can often be found in a data dictionary but may also be embodied in programs, production rules or user protocol. They are 'world' constraints in that they define relationships and domains of the information contained in the files. They ensure that the data remain consistent with a defined view of the world. Within such an information system it is recognized that there exist facts that normally remain unchanged because they are absolute statements of the world: facts such as the range of possible colours, characteristics of drugs or $2+2=4$. There are also facts that represent the dynamic behaviour of the world. It is through the world constraints that the dynamic facts affect and change the data in the file.

The third type of system and the first category of expert system includes fact assessment, which calls upon special assessment procedures to quantify the value of data retrieved and propose new retrieval tasks from the results. IR has concentrated on fact assessment, a technique that is central to expert systems particularly when using fuzzy methods (Karavnou, 1982). In CRIB new retrieval tasks are controlled by the user; in RAFFLES they are predefined and in MYCIN they can be either controlled by the user or automatically triggered by the current circumstances. The dynamic facts of CRIB and MYCIN are the particular observations made at the time of diagnosis. However, in MYCIN such facts are structured.

The fourth type of system and the second category of expert system provides rules for finding facts. The database for a particular individual is created according to a prescription and guided by world constraints. MYCIN assumes that patients are different and the rules for accumulating facts about a particular patient are provided so that a unique and structured database concerning that patient is produced. It is then that the assessment procedures are used to produce a recommended therapy.

## 7. UNCERTAIN FACTS, INEXACT REASONING AND LINGUISTIC HEDGES

The requirement of expert systems to provide fact interpolation is part of the reason for having a fuzzy filing system. The fourth type of system, as described, would generate a factual database about a particular patient (or prospective geological site) and it would be up to the assessment procedures interacting with both the fixed facts of symptoms (or the geological indicators) and drug properties to produce the appropriate interpolation. Observed facts, which do not always tally exactly with the specification of facts predefined by the system, can be potential members of several specifications. This problem arises because most of the descriptions of the world made by people depend upon linguistic performance and normative values. It is tempting to suggest that people are being inexact, vague, woolly or fuzzy but the language employed merely reflects the world as it is perceived. To say 'the girl was near me' is a more appropriate statement of fact than 'the nearest point of contact of the girl was 5.362 cm.' In the repair and maintenance of man-made machines most facts can be given unequivocally but in the treatment and care of people or the divining of ores from geological observations the facts cannot be so precise.

The *essence* of the method that can cope with inexact facts within a computer program can be found in fuzzy set theory. In MYCIN a technique has been developed under the title of 'a model of inexact reasoning'. This model uses equivalent equations and performs a comparable task to that of fuzzy set theory (Zadeh, 1965). The membership values are interpreted as certainty factors, which are further constrained. For example, an object could be described as oval, round, square or rectangular in appearance with no certainty that it is any particular shape. However, it may be more oval than round and more rectangular than oval. This indistinction of category can be quantified by assigning a number between 0 and 1 to each description. Thus the object's shape might be described as ((rectangular, 0.6) (oval, 0.3) (round, 0.1) (square, 0.0)) where the numbers indicate some measure of certainty of each classification. Since the object has to have some shape then the assigned numbers are expected to be not greater than one. However, if these numbers are membership values then this restriction is not necessary. So unlike a normal database an attribute can be multivalued and indecisive.

Objects are not normally identified according to a single attribute and even if each attribute had a definite value the classification of such an object can also be indistinct. Linear equations are provided with arbitrary thresholds which allow the uncertainty of attributes describing an indistinct object to arrive at some useful decision. Objects for MYCIN are bacteria whose attribute values are derived from both clinical and laboratory data. The nature of the problem domain is such that supportive evidence for one bacteria does not preclude the existence of others.

Natural language provides tools for making subtle distinctions along attribute dimensions. A colour can be simply 'red' but it may also be described as a sort of orange-red. This state of affairs may be represented within the computer program as (Colour(Red 0.7) (Orange 0.3)) or (Colour 6000) where 6000 is the wavelength. The term 'sort of' can be predefined for each colour but the range of possibilities can be extremely large. There is a technique described by Efraim Shaket (1976) employing a membership function that converts physical values to certainty (membership) values. These values can then be modified by transformations associated with linguistic hedges—such as 'very', 'too', 'rather', 'most', 'less', 'sort of', etc.—which effectively cause a shift in fuzzy set values to accord with human expectations. However, it has been suggested that using an assessment of hedges by adding up the pros and cons could be as effective as attempting to produce a numeric model of English descriptions (Fox, 1981).

## 8. EXTENDED RELATIONAL ANALYSIS

In order to make a more detailed comparison between the different types and categories of systems it is now necessary to introduce an extended form of relational analysis. The problem to be overcome is that each different type of system has been described at a detailed level within its own cartel. The advantages of extended relational analysis are that it can describe world constraints in a form that is simple, unique and independent of the storage or program structures. It is through extended relational analysis that the similarities and differences of systems can be highlighted.

The cardinal principle on which extended relational analysis depends is 'functional dependence'. Functional dependence derives from mappings of the kind:

$$y = f(x)$$

where the value of '$y$' is dependent upon the value of '$x$' via some mapping function

'$f$'. In general $f$ has the property that there can be many values of $x$ that map into a single $y$, or there may be a single one-to-one mapping of $x$ into $y$. There can never be a value of $x$ that does not produce a value $y$ although there may well be values of $y$ with no corresponding $x$. Another way of expressing this cardinality characteristic is by the symbol $\succ\!\!-\!\!-$ so that:

$$x \succ\!\!-\!\!- y$$

represents the statement that $y$ is functionally dependent upon $x$ via some unspecified mapping function. It is unspecified because only the cardinality relationship just described is required which holds for all the mapping functions being considered; $x$ in this case is called the 'determinant'. This form of functional dependency is sometimes called 'O-implies' or simply 'implies' although it is only loosely connected to the logical 'implies' (however, *see* Fagin, 1977).

There are other mapping functions that may be more constrained in that there can never be values of $y$ with no corresponding $x$, although $x$ may map in the many-to-one or one-to-one mixture as before. This class of mapping function can be expressed by the symbol $\triangleright\!\!-\!\!-$ so that:

$$x \triangleright\!\!-\!\!- y$$

represents the statement that $x$ 'determines' $y$ and that there is no value of $y$ without at least a single value of $x$ associated with it. This is often referred to as '1-implies' although it is *not* a logical implication.

The functions described so far could be continuous and typically some kind of polynomial but the importance of this reduced specification of the relationship between $x$ and $y$ is where $x$ and $y$ represent some finite set of individuals. An element $x$ of $X$ determines an element $y$ of $Y$ by the specification

$$X \succ\!\!-\!\!- Y$$

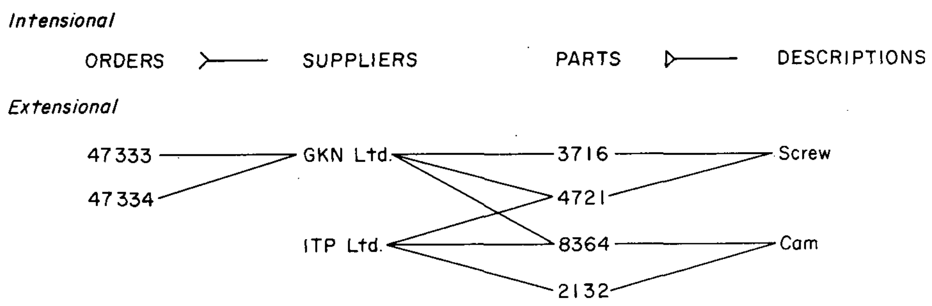


FIG. 3. The functional dependencies between attributes of a database

Figure 3 illustrates both kinds of functional dependence between the attributes ORDERS, SUPPLIERS, PARTS and DESCRIPTIONS. The mappings defined are 'intensional' in that they specify what can be allowed but not necessarily what is current. The current values are the 'extensional' part of the description. Figure 3 states that every order uniquely identifies a supplier but not all suppliers have orders going to them. However, all parts have one and only one description but some descriptions are used more than once. On the other hand there is no functional dependency between suppliers and parts since some parts are sold by several suppliers and some suppliers sell several parts.

Functional dependency has been somewhat laboured because it will be used as the main descriptor of world constraints and it is these world constraints that define a model of the data that is homomorphic with respect to user requests. The other important descriptor is the normalized relation. A relation is written as a named list of attributes such as:

PART (PartNo: Description, Quantity)
PARTSUPPLIER (PartNo,SupplierNo: Price, CodeNo)
SUPPLIER (SupplierNo: Address, Name, Status).

The attributes within these normalized relations can be grouped either as 'keys' or 'owns'. Keys are terminated by a colon and it is possible to have more than one key grouping in a normalized relation. If there is more than one attribute to a key as in PARTSUPPLIER, then the key is 'composite' and is treated as though it were a single attribute. Own attributes are written to the right of all the keys. A relation is normalized (Boyce/Codd normal form) if every determinant is a key. There are other normal forms but this is the only normal form required for this paper. Figure 4 illustrates the normalized relations exposing the functional dependencies between key and own attributes.

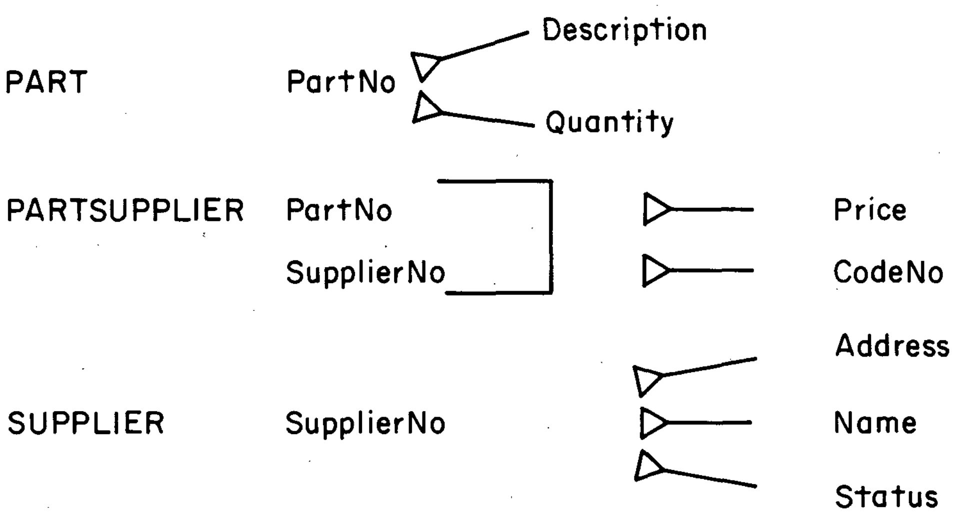


FIG. 4. The $A$-functional dependencies of relations in Boyce/Codd normal form

The description of relational analysis so far consists of well accepted ideas apart from the introduction of the two types of functional dependencies and their respective symbols. The reason for this finer distinction becomes clear when the functional dependencies are used for reconstructing the normalized relations into a complete description of the constraints of the world represented by the data under analysis. A relation can be considered as a set of tuples where each element of the set is identified by its key. As with any set of sets, mappings can be defined if they exist. Consequently, it is possible to have one relation functionally dependent upon another. In the previous example the relation PARTSUPPLIER is a determinant of the relations PART and SUPPLIER. Any tuple in the PARTSUPPLIER relation can uniquely identify a tuple in the PART relation as well as in the SUPPLIER relation. In this case it is possible to have tuples in either the PART or the SUPPLIER relations which do not have any corresponding tuple in the PART-SUPPLIER relation. Thus the less constrained functional dependency is used.

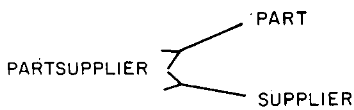


FIG. 5. The *R*-functional dependency graph for PARTSUPPLIER, PART and SUPPLIER

Figure 5 shows the functional dependency graph for this example. To distinguish between the levels of description of Figures 4 and 5 these graphs will be called *A*-functional dependency graphs and *R*-functional dependency graphs respectively.

The importance of this reconstruction is threefold:

1. It describes precisely what relations have to be checked during any update. Thus if a new PARTSUPPLIER tuple is to be introduced then both PART and SUPPLIER must be scanned for the existence of the appropriate PartNo and SupplierNo to be used. Alternatively, if a SUPPLIER tuple is to be deleted then any PARTSUPPLIER tuple dependent upon it must also be deleted. The process of changing a value of an attribute depends upon whether it is a key or own attribute. Key attributes, by their nature, require the deletion and reinsertion of a tuple but own attributes normally can simply be changed although they may cause deletion and reinsertion elsewhere.
2. It forms the basis of any file design decisions since any physical distribution of the data ought to maintain the dependencies given.
3. It provides a primitive model for an interactive language understanding system (Addis, 1982a). The user merely has to supply the attribute names to be retrieved and the attribute value pairs connected logically to form selection criteria on those names for the system to generate the appropriate sequence of file operations to perform the retrieval task. Further constructive statements linking different functional dependencies can be introduced that define other restrictions which are described elsewhere (Addis, 1982b). However, these additional extensions are not required for the comparisons to be made in this paper.

The description of the constraints of the world under consideration requires that the attributes be 'typed'. The attributes in the relations are drawn from a set of 'domains' or ranges of values. Thus, a colour attribute may refer to a limited set of colours such as (red, orange, yellow, green, blue, indigo, violet, black, brown, white) and any colour outside this range is considered not a colour. Days of the week, months in the year are other familiar fixed domains of values. In some cases a function can be specified which will represent the set of acceptable values for a particular domain. A function is necessary for infinite sets such as dates or even numbers. Order numbers may be expressed as being a four-figure number or part numbers may always need to have a letter prefix. Domains are normally considered to be fixed, but it may be required, for example, to reduce the choice of values every time a value is chosen. This particular requirement is usually satisfied by creating a relation with an attribute whose values are drawn from the domain, and this attribute is the candidate key.

In some cases an update constraint needs to be specified which restricts the ability to delete or add tuples to a relation. This restriction is marked by prefixing a relation ($R$) with symbols where:

- $-R$ indicates a relation $R$ which can only be deleted from. This represents an irreplaceable resource that is drawn upon such as allowed order numbers or the paintings of Constable.
- $+R$ is a relation $R$ which can only be added to and represents an accumulation of entities. Such a restriction may be required in order to prevent inconsistencies in that the deletion of a tuple can generate unwanted effects throughout a model and it may be more appropriate to assign a termination attribute which makes those tuples no longer needed.
- $\#R$ shows that relation $R$ can be neither deleted from nor added to although there may be no restriction on changing the value of the own attributes.

Relations which are unmarked are assumed to be completely updatable and have no restrictions other than that imposed by their position in the relational model.

A complete set of world constraints for a commercial database can now be represented by a functional dependency graph, list of normalized relations and domain specifications. Provided the implementation remains homomorphic to this specification then the dynamic behaviour of the system can be understood and followed. This has the great advantage of avoiding unnecessary implementation details which often obscure the processes involved. An example of such a type-two system is the ICL Purchasing System 1970, which is illustrated in Figure 6.

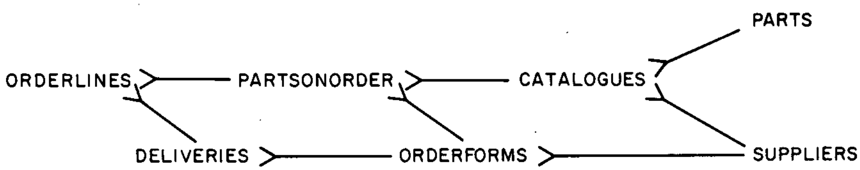


FIG. 6. $R$-functional dependency graph for a purchasing database. A type-two system

The functional dependencies declare a set of truths that must be maintained by update programs. Thus, if a new catalogue item is to be inserted then a part in PARTS and a supplier in SUPPLIERS must also exist to correspond with that item. On the other hand if an item is to be deleted from the CATALOGUES relation then all elements in PARTSONORDER and ORDERLINES that imply that CATALOGUES'item must also be deleted. If a query is made to the database that only involves attributes of PARTS and SUPPLIERS then the linking relation CATALOGUES must be used to fulfil the request because it represents the least redundant relation. This least redundant relation is then used as the base for joining with the implied relations in order to produce a compound solution that contains all the required information. These relational operations can then be translated into a sequence of search tasks on the files. In this manner complex retrieval procedures can be generated from what is to the user a simple request (for more detail *see* Addis, 1982a).

## 9. CRIB—A TYPE-THREE SYSTEM

The reason for developing CRIB was because modern hardware and software are becoming more complex and are demanding a much greater level of diagnostic skill and competence from engineers and system experts. However, as systems proliferate, highly qualified and experienced personnel will become less available. Apart from developing more reliable systems, a solution to this shortage is to reduce the time spent by the skilled in finding faults. This can be achieved by providing better fault-locating aids so that either the more competent can find faults quicker or the less competent (more available) can locate faults which would otherwise require greater skill. Compounding this requirement for great skill is the need for rapidly communicating new modifications of systems and fault-locating techniques gleaned from others' experience.

CRIB (Computer Retrieval Incidence Bank) was devised as a means of accumulating information on critical symptoms used for diagnosis by an expert engineer in the field and using this information to help his colleagues. This idea of accumulating fault reports for computers is not new and was used in the early 1960s for the ATLAS computer at Harwell. What was new in the CRIB system was both the use of world constraints to maintain a database of diagnostic reports and the extensive use of 'action' assessment procedures to help the user find the fault.

The CRIB system consisted of two programs. The first program was an update and general enquiry program (Addis, 1982a; Babb, 1982) that used the CAFS as its main means of data storage and access. The essential property of this first program is that it used an internal representation of the $R$-functional dependency graph to maintain consistency and interpret queries without any need of the user to be aware of the relational structure. The second program retrieved data from the CAFS and assessed the next best tests to be done in order to rapidly home in on the current fault.

A fault was considered to be both a replaceable unit and a set of symptoms where a symptom was a test with a result. A test could be as complicated as carrying out some procedures and measurements or as simple as making an observation. For any particular fault the symptoms could be grouped together as either:

1. A total group of symptoms where all observations concerning a fault are accumulated (TGSYMP).

2. A key group of symptoms which is the sufficient set of symptoms to isolate the fault (KGSYMP).
3. A sub-group of symptoms which is the necessary set of symptoms associated with the fault (SGSYMP).

The set of symptoms gleaned during a diagnostic session represents the changing facts. The fixed facts are the structure of the machine in terms of a hierarchy of replaceable units and various symptom groups accumulated as new information and experience come in from the field. Thus a key group may isolate the fault as a peripheral but the particular peripheral will require additional symptoms to be identified. The structural description extends only to atomic replaceable units but if CRIB fails to discriminate to this level then the engineer will at least have arrived at a reduction of the problem. The *R*-functional dependency graph for CRIB is shown in Figure 7.

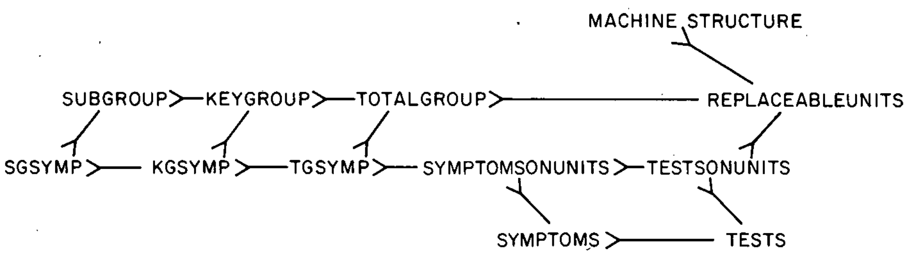


FIG. 7. *R*-functional dependency graph for the CRIB diagnostic aid. A type-three system

The initial symptoms provided freely by the engineer are used to form the key group that will be the most discriminating (deepest point in the structural hierarchy). The tests that will provide further detailed discriminations are retrieved and ordered according to their power to refine. Tests are quantified by a function whose parameters are 'time to do the test', 'percentage of faults that use the test' and 'probable effectiveness of test from past results'. A selected list of the most informative tests are presented to the engineer who may either choose one or independently do a test not recommended.

An example of the protocol is given below where the engineer's responses are underlined. The responses of the program have been abbreviated and comments have been made in brackets to make clear the process.

<u>Alter and display ones ok</u> S0402. Any more?
<u>Program load from disk not ok</u> N0022. Any more?
<u>No</u> (Two symptoms accepted, CRIB starts search and assesses results.)

Match in CPU. Suggested actions:

* Check parity error—A0008
* Load MPD 2 LOG move—A0415

* Run WAMS—A0389
* Check run light—A0413

(The four most effective tests (actions) have been selected.)

<u>S0 N0008, N0415, S0413</u> (The engineer can use the code numbers prefixed with S or N for OK or not OK depending on the result of the test.)

Any more?

<u>NO</u> (second cycle begins)

Match in CPU. Suggested actions:

* Change board 9714—A0405
* Check hand switches indirect—A0418
* Change board 9716—A0421
* Check steps of disk IPL ROM—A0406

<u>Log out (S0405) OK</u> (In this case first action clears fault.)

The problem with the jargon English interpreter was that the engineer could never be sure how to phrase a sentence so that it would be accepted or if accepted whether it had been identified correctly. However, the engineers soon learnt a few starting sentences and then employed the returned action codes with the appropriate prefixes to continue. The precompiled RAFFLES system presumes this naturally occurring strategy throughout a diagnostic session (Addis, 1982c).

## 10. MYCIN—A TYPE-FOUR SYSTEM

The world constraints, as represented by the *R*-functional dependency graph, are used both for maintaining the relative truth conditions of the stored data and for generating search tasks from user-specified requests. A type-four system is concerned with gathering facts in order to produce a current statement of the world as perceived by the user and to gather the facts correctly requires update protocols synthesized from a model of the relationships between the facts. MYCIN retains fixed facts about bacteria and the symptoms normally associated with them. MYCIN also has information about drugs and what sort of bacteria they can affect as well as potential side effects of the drug on the average patient. What MYCIN has to construct is a model of the patient and this model is in the form of a database confined to particular update properties. Once the model of the patient has been achieved MYCIN can then call upon special assessment procedures which use the characteristics of the patient defined within the database to recommend a therapy consisting of a minimum set of drugs.

The primary task of MYCIN is to determine what significant organisms exist within the patient. To achieve this task there are two kinds of production rules of the form:

IF <premise> THEN <action/conclusion>

The first kind of rule is concerned simply with the problem of collecting data from the consultant using the system. These data may well be results of laboratory tests or observations made by the consultant. The second kind of rule is used to maintain data consistency. The update rules are normally described as further IF . . . THEN rules but they can also be represented by an *R*-functional dependency graph. The advantage of the latter is that it provides a uniform method of description that can be displayed in a single figure.

Rules of both kinds can have premises which are indistinct (fuzzy) and associated with the interpretation of these rules is a procedure that can calculate a value representing the certainty of the conclusion (action). Premises consist of a list of conjoined conditions and these conditions are concerned with statements of facts such as 'name of patient', 'identity of organism', 'drug allergies', 'locus of infection', 'result of a test'. Each of these facts can have a range of certainty that must be satisfied in order to be considered true. Examples are given in Tables 1 and 2.

Table 1. Examples of fuzzy functions that do not form conditionals but describe attribute status

| | | *Fuzzy function* | | *Context* | | *Parameter* | *Certainty range* |
|---|---|---|---|---|---|---|---|
| 1. | It is | definite | that the | patient | is | Jones | +1.0 |
| 2. | It is | known | that the | organism | is | streptococcus | +0.2 to +1.0 |
| 3. | It is | not definite | that the | site | is | blood | —1.0 to +1.0 |

Table 2. Examples of fuzzy functions that control conditional statements on clinical parameters

| | | *Context* | *Attribute* | | *Fuzzy relation* | | *Value* | *Certainty range* |
|---|---|---|---|---|---|---|---|---|
| 4. | The | organism-1 | morph | is | same | as | rod | +0.2 to +1.0 |
| 5. | The | organism-3 | Gram | | definitely | is | Gramneg | +1.0 |
| 6. | The | organism-4 | air | | might-be | | aerobic | —0.2 to +1.0 |

Rules can be constructed using the kind of fuzzy functional statements given in Tables 1 and 2 thus:

*IF* The organism-1 Gram is same as Gramneg
and morph is same as rod
and air is same as aerobic

*THEN* Conclude that organism-1 classification is Enterbacteriaceae
with a certainty of +0.8

If the database on the patient contained the information shown in Table 3, then the result would be:

(MINIMUM (1.0, 0.8, 0.6))

and this is *true* because it is in the range (+0.2 to +1.0) which tallies with the fuzzy function requirement 'same'.

Table 3. An example of a fuzzy database with multiple values

| *Context* | *Parameter* | *Value* |
|---|---|---|
| Organism-1 | Gram | ((Gramneg 1.0)) |
| Organism-1 | Morph | ((Rod 0.8) (Coccus 0.2)) |
| Organism-1 | Air | ((Aerobic 0.6) (Facul 0.4)) |

The result of 0.6 now interacts with the certainty factor of the conclusion by simple multiplication giving a value of 0.48. There are three English translations of the positive value (the negative value simply prefixes the translation term with *not*) depending on the range of this certainty value:

Strongly suggestive (+0.8 to +1.0)
Suggestive (+0.4 to +0.8)
Weakly suggestive (+0.0 to +0.4)

The above example could thus cause the program to respond with the tentative hypothesis:

There is suggestive evidence that organism-1 is Enterbacteriaceae (0.48).

Further evidence found by other rules for the same hypothesis can also modify this result by another simple process:

$$C1 + C2*(1 - C1) = \text{new certainty factor for hypothesis}$$

where C1 and C2 are the certainty factors of the final conclusions of the two interacting rules about the same hypothesis.

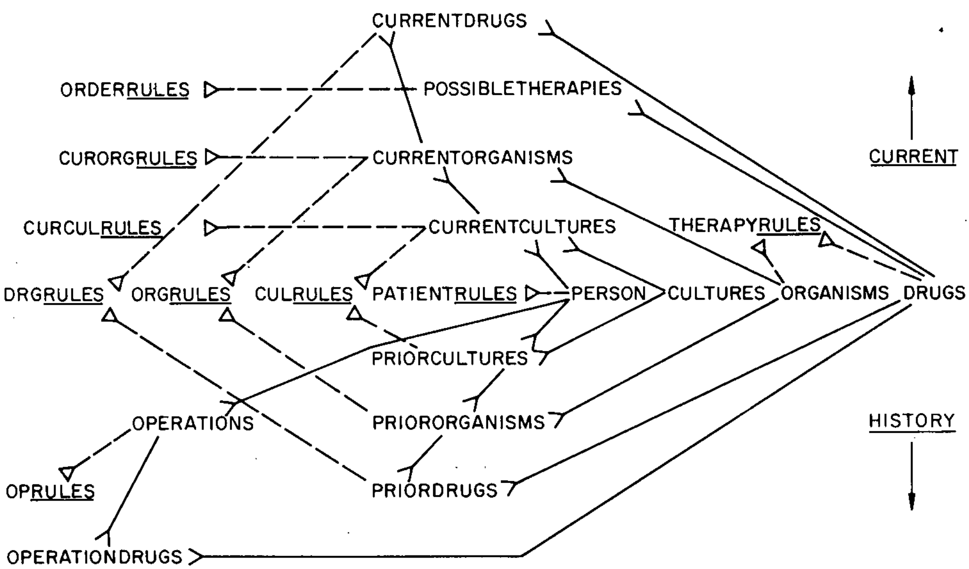


FIG. 8. *R*-functional dependency graph for MYCIN. A type-four system

The functional dependency graph shown in Figure 8 was deduced from the book *Computer-based Medical Consultations: MYCIN* by E. H. Shortliffe. The descriptive information was incomplete and hence various dependencies had to be assumed. The interesting addition to database descriptions in the form of *R*-functional dependency graphs is the rules associated with some of the relations. These rules are shown at the end of a dashed l-implies which is an abbreviation in this paper for the structure of a pair of double tied dependencies as shown in Figure 9.

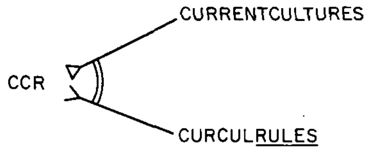


FIG. 9. Part of the *R*-functional dependency graph showing the correct description of the rule dependency

When two relations such as CURRENTCULTURES and CURCUL*RULES* are double tied together then the relation CCR at the root of the double tie represents a set of tuples that map one to one with a subset of the natural join of a subset of these two relations. In this case every rule is associated with its appropriate culture within CCR. The existence of a l-implies as one of the dependencies shows that every culture must trigger the appropriate set of rules (Addis, 1982b).

The *R*-functional dependency graph shown in Figure 8 with the associated set of rules describes a system that controls the creation of a structured database about an individual. In addition to these rules for maintaining structure (world constraints) there is a further set of rules concerned with collecting information about the individual. The control of the process of collecting data flows back and forth between the two sets of rules and the process of IR now becomes combined with update procedures. The primary data collecting rule is rule 092 which states

IF: (1) There is an organism which requires therapy.

and

(2) Consideration has been given to the possible existence of additional organisms requiring therapy, even though they have not actually been recovered from any current cultures.

THEN: Do the following:

(1) Compile the list of possible therapies which, based upon sensitivity data, may be effective against the organisms requiring treatment.

and

(2) Determine the best therapy recommendations from the compiled list.

OTHERWISE: Indicate that the patient does not require therapy.

In order to satisfy the premise of rule 092 other rules that have the conclusion:

'There is an organism which requires therapy'

are extracted and applied. In this case there is only one rule (rule 090) which satisfies this requirement and this in turn leads to other rules which finally need data to be collected about the patient. When this occurs, the rules concerned with maintaining the world constraints are triggered since information about organisms cannot exist until there are cultures (*see* Fig. 8). Further, cultures cannot occur unless there is a person and the first thing required to be known about a person is his name, age, etc. MYCIN will thus print the relation name being created followed by an *n*-tuple number which becomes an internal identifier of this *n*-tuple (key). This is then followed by a request for attribute values. The user's responses are in italics.

———PATIENT-1———
1. Name: *J. Sample*
2. Sex: *Male*
3. Age: *60*

Other constraint rules are activated as the database under construction checks consistency and demands information from the user or searches its own database to maintain it. Once all the database requirements are satisfied then the data collecting rules are allowed to proceed. Thus MYCIN introduces four important additions to an information system:

1. Weighted multiple values of attributes.
2. Additional update rules associated with relations.
3. Meta-rules concerned with database creation.
4. An interrogation and search procedure that involves the construction of a model of the problem (patient) under consideration.

When MYCIN has satisfied all the consequences of the premise of rule 092 then data assessment procedures are activated (as in CRIB) to compile a list of possible drug combinations that will attack the organisms and are suitable for the patient.

## 11. CONCLUSIONS

Expert systems were derived mostly from work that was mainly concerned with emulating human intellectual behaviour. The reason for this concern was to understand and study the mechanics of human thought processes. From this study it was believed that machines could be created that would have all the advantages of intelligence and insight without the disadvantages of forgetfulness and error that plague humanity. The expert system was thus meant to represent the complete and ideal expert within a narrow field of knowledge who would not forget, draw illogical conclusions (although they may be mistaken), be emotionally biased or need to sleep.

Several expert systems have been examined and it was concluded that there were five different types. The differences between these types were the extra enhancements added to a computer information retrieval system (*see* Table 4). The basic information retrieval system (type-one system) was considered to be capable of storing fuzzy data in that attributes could have weighted multiple values. Not all types of system used this facility.

Table 4. Enhancement table

| *System type* | *Expert system categories* | *Enhancements* |
|---|---|---|
| 1 | 0 | Files with online access. The information may be fuzzy, i.e., weighted multiple values |
| 2 | 0 | World constraints maintained automatically by update procedures |
| 3 | 1 | Fact assessment procedures and mechanisms to aid fact retrieval |
| 4 | 2 | Rules for collecting new facts |
| 5 | 3 | Mechanisms for generating new rules that can modify the world constraints (new model), adjust assessment procedures and provide new rules for collecting facts about different hypotheses |

The expert systems are best viewed as information retrieval aids for experts that can involve update procedures and are a method of communication of group practices rather than acting as a simulated expert (Addis, 1981). The major sources of confusion that can occur if expert systems are considered equivalent to a human expert are the distinctions between:

1. Naturally occurring performance.
2. The formal representation of an abstraction of that performance.
3. The simulation of such a representation on a computer.
4. Agreed ideal behaviour or practices.

Most artificial intelligence systems simulate the agreed ideal behaviour but there is a trend towards simulating the abstractions of naturally occurring performance as models of human cognition. Within the realm of expert systems this trend is taken in order to ensure that human decision makers can follow arguments that support any proposals made by the system (Michie, 1979). In this way the responsibility of any action based on these proposals can remain squarely upon the shoulders of the man. However, this trend weakens the usefulness of expert systems in that it restricts the strategies of problem solving to those within the user's intellectual grasp.

An alternative view can be taken that considers the expert system as a source of mutually agreed competence that is capable of providing human solutions to problems. However, although these human solutions may often be beyond the capability of any individual, the user will always be in the position to decide the relevance of such mutually agreed competence and should thus be placed in a position of controlling the search through an implicit maze of potential solutions.

## REFERENCES

Addis, T. R. (1980) Towards an 'expert' diagnostic system. *ICL Technical Journal May*, 79–105.

Addis, T. R. (1981) *Applied epistemics: an empirical view of knowledge communication.* Brunel University: Man-Computer Studies Group (CS/MMSG 2).

Addis, T. R. (1982a) A relation based language interpreter for a content addressable file store. *ACM Transactions on Database Systems 7*, 125–163.

Addis, T. R. (1982b) *Relational analysis for artificial intelligence.* Brunel University: Man-Computer Studies Group (CS/MMSG 8).

Addis, T. R. (1982c) Knowledge refining for a diagnostic aid (an example of man–machine epistemics). *International Journal of Man–Machine Studies 17.* (In press)

Babb, E. (1982) Joined normal form: a storage encoding for relational databases. *ACM Transactions on Database Systems.* (In press)

Bennet, J. S. *et al.* (1979) *Applications-oriented AI research: science and mathematics.* Stanford Heuristic Programming Project (Memo HPP-79-22).

Buchanan, B. G. and Feigenbaum, E. A. (1978) DENDRAL and Meta-DENDRAL: their applications dimension. *Journal of Artificial Intelligence 11*, 5–24.

Davis, R. and Buchanan, B. G. (1977) Meta-level knowledge: overview and application. IJCAI, 920–928.

De Dombal, F. T., Leaper, D. J. and Horrocks, J. C. (1974) Human and computer-aided diagnosis of abdominal pain: further report with emphasis on performance of clinicians. *British Medical Journal 1*, 376–380.

Dreyfus, H. L. (1972) *What Computers Can't Do (A Critique of Artificial Reason).* New York: Harper and Row.

Duda, R., Gaschnig, J. and Hart, P. (1980) Model design in the PROSPECTOR consultant system for mineral exploration. In *Expert Systems in the Micro-electronic Age.* (D. Michie, ed.) Edinburgh University Press.

Duda, R. O., Hart, P. E. and Nilsson, N. J. (1976) Subjective Bayesian methods for rule-based inference systems. *AFIPS 45*, 1075–1082.

Edwards, W. (1972) N = 1: Diagnosis in unique cases. In *Computer Diagnosis and Diagnostic Methods.* (J. A. Jacques, ed.) pp. 139–151. Springfield, Ill: Charles C. Thomas.

Fagin, R. (1977) Functional dependencies in a relational database and propositional logic. *IBM Journal of Research and Development November*, 534–544.

Feigenbaum, C. A. (1980) Themes and case studies of knowledge engineering. In *Expert Systems in the Microelectronic Age.* (D. Michie, ed.) pp. 3–25. Edinburgh University Press.

Fox, J. (1981) Statistical and non-statistical inference in medical diagnosis. February 1981, draft document.

Fox, J., Barber, D. and Bardham, K. D. (1980) Alternative to Bayes? A quantitative comparison with rule-based diagnostic inference. *Methods of Information in Medicine 19*, 210–215.

Gough, P. B. (1966) The verification of sentences: effects of delay of evidence and sentence length. *Journal of Verbal Learning and Verbal Behaviour 5*, 492–496.

Hartley, R. T. (1982) *The competent computer.* Brunel University: Man-Computer Studies Group (CS/MMSG 6).

Johnson, L. (1980) Practical reasoning. *Progress in Cybernetics and System Research 8.*

Johnson, L. (1982) *Epistemics and the frame conception of knowledge.* Brunel University: Man-Computer Studies Group (CS/MMSG 7).

Karavnou, E. T. (1982) *Information retrieval with fuzzy logic extension.* Project report for the degree of B.Tech. Brunel.

Lenat, D. B. (1977) The ubiquity of discovery. *Journal of Artificial Intelligence 9*, 257–286.

Michie, D. (1979) Problems of the 'human window'. In *Expert Systems in the Micro-electronic Age.* (Presented at the AISB Summer School on Expert Systems in Edinburgh, July)

Nordyke, R. A., Kulikowski, C. A. and Kulikowski, C. W. (1971) A comparison of methods for the automated diagnosis of thyroid dysfunction. *Computer Biomedical Research 4*, 374–389.

Oddy, R. N. (1977) Information retrieval through man-machine dialogue. *Journal of Documentation 33*, 1–14.

Pauker, S., Gorry, G. A. and Kaffirer, J. P. (1976) Towards the simulation of clinical cognition—taking a present illness by computer. *American Journal of Medicine 60*, 981–996.

Pople, H. E. (1977) The formation of composite hypotheses in diagnostic problem solving—an exercise in synthetic reasoning. *IJCAI 5*, 1030–1037.

Quinlan, J. R. (1979) *Induction over large data bases.* Stanford Heuristic Programming Project Memo HPP-79-14 (Report No. STAN-CS-79-739).

Sacerdoti, E. D. (1974) Planning in a hierarchy of abstraction spaces. *Artificial Intelligence 5*, 115–135.

Shaket, E. (1976) Fuzzy semantics for a natural-like language defined over a world of blocks. *Artificial Intelligence Memo 4* (University of California at LA).

Shortliffe, E. H. (1976) *Computer-based Medical Consultations, MYCIN. Artificial Intelligence Series 2.* Amsterdam: Elsevier.

Sparck Jones, K. (1973) Index term weighting. *Information Storage and Retrieval 9*, 619–633.

Sparck Jones, K. (1974) Automatic indexing: a state of the art review. *Journal of Documentation 30*, 393–432.

Toulmin, S. (1976) *The Uses of Argument.* Cambridge University Press.

Zadeh, L. A. (1965) Fuzzy sets. *Information and Control 8*, 338–353.