NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Enginnering

EXPERT SYSTEMS FOR MEDICAL DIAGNOSTICS

Graduation Project
COM400

Student: Derviş Koray Uğural (93394)

Supervisor: Assist. Prof. Dr. Rahip Abiyev

Lefkoşa – 2000

CONTENTS

NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Enginnering

EXPERT SYSTEMS FOR MEDICAL DIAGNOSTICS

Graduation Project
COM400

Student: Derviş Koray Uğural (93394)

Supervisor: Assist. Prof. Dr. Rahip Abiyev

Lefkoşa – 2000

CONTENTS

ACKNOWLEDGEMENT

I would like to express to graduate for my supervisor Assist. Prof. Dr. Rahip Abiyev for his invaluable and help during the course of graduation's degree.

Also my special thanks goes to Faculty of Engineering Dean Prof. Dr. Khalil Ismailov, Department of Computer Engineering Chairman Assist. Prof. Dr. Adnan Khasman, Assistant to Dean for Academic Affairs Tayseer Alshanableh and Staff of Computer Engineering, where their advise and instructions was very helpful for me to finish graduation. Many thanks for the Vice-President of Near East University Associate Prof. Dr. Şenol Bektaş for giving me the opportunity continue my graduation.

I would like thanks to my parents, my brother, my sister and my friends for their supports and their patiences.

ABSTARCT

In this graduation project, the development of expert system for medical diagnostic is considered. To solve this problem the followings are performed. The expert system concept, its main characteristics, the compressions of expert system with decision support systems (DSS), and also the descriptions of different expert systems are given. After the structure of expert system and the functions of its main blocks are described. Models of knowledge representation, such as OAV triplets, semantic networks, predicate logics, frames, neural networks, rule-based systems are described. To solve given problem the rule-based model is chosen, their main properties are widely described. After the analysis of knowledge acquisition and their realization are considered. As an example, the development of diagnostics expert system for stomach and intestine diseases is considered. Using experienced expert knowledge and different medical references the knowledge-based is created. This knowledge-based has about 256 production rules. Premise part of rules includes the input features of stomach diseases, and the conclusion part includes diagnosis. The considered expert system is realized on the base of ESPLAN expert system shell.

INTRODUCTION

During the past decade there has been a virtual explosion of interest in the field known as expert systems (or, alternatively, as known knowledge-based systems). In fact, some cynics might say that it is becoming harder and harder to find problem for which expert system has not been propesed. And it would seem that no computer software support system can not be considered complete without the inclusion of at least one expert system development package, or "shell". While such intense hype resulted in considerable misunderstanding, and even misuse, of the expert system methodology. Instances of the implementation of the expert systems by the wrong people, in the wrong manner, and on the wrong problem are, more common than one might expect. This is unfortunate as it serves to obscure the irrefutable ability and potential of expert systems in decision making, including decision making in the host of important, real world situations for which there are simply no appropriate alternatives.

The construction of an expert system is relatively straightforward procedure, requiring very little in the way of background in the sohisticated mathematical methods absolutely none in computer programming. The development and implementation of expert systems for real-world require time, training, and experience, plus a certain natural anffinity for dealing messy, ill-structed problems. And first and foremost, one must select a problem for which implementation of expert system is appropriate. This implies, most strongly, that one must also be aware of potential alternatives to the expert systems approach. Those who identify appropriate applications of expert systems and who perform the process of development and implementation are called knowledge engineers.

Traditionally, new product introduction is accomplished in series of steps, utillizinginput from numerous individuals having expertise in variety of areas. In the terminology of expert systems, these human experts are called domain experts, each posseses expertise in a very specific, and relatively narrow domain. The domain experts serve in te following manner:

- To select, from amount of a new list of canditatenew products, those that would seem to hold the most promise
- To develop, for each new product selected, a plan for the introduction of that product on the trail basis
- To use the results of the test maketting efforts to decide which product will finally be incorporated into the firm's product line, how to initiate and control the manufactureof these products, distribute and advertise them how to price the products, and so on

CHAPTER 1

APPLICATION OF EXPERT SYSTEM FOR SOLVING DIAGNOSTIC PROBLEMS

1.1. THE EXPERT SYSTEM CONCEPT

The expert system is a recent addition to a circle information systems. Expert systems are computer based systems that help managers resolve problems or make better decisions. However, expert systems, which are also referred to case-based reasoning systems, do so with decidedly different twist. An expert system is an interactive computer based-system that responds to questions, asks for clarification, makes recommendations, and generally helps the user in the decision-making process. In effect, working with an expert system is much like working directly with human expert to solve a problem. It even uses information supplied by a real expert system in a particular field such as medicine, taxes, or geology. Expert systems re-create the decision process better than humans do. We tend to miss important considerations or alternatives – computers don't.

An expert system applies preset IF-THEN rules to solve a particular problem, such as determining a patient's illness. Like management information systems and decision support systems, expert systems rely on factual knowledge, but expert systems also rely heuristic knowledge and the heuristic rules of thumb used in an expert system are acquired from a real live domain expert system, a human expert in a particular field, such as jet engine repair, life insurance, or property assessment. The expert system uses this human-supplied knowledge the human thought process within a particular area of expertise. Once completed, an expert system can approximate the logic of a well-informed human decision maker.

An expert system is a computer program that represents and reasons with knowledge o special subject with a view to solving problems or giving advice.

An expert system may completely fulfill a function that normally requires human expertise, or it may play the role an assistant to human decision maker. In order words, the client may interact with the program directly, or interact with human expert who interacts with the program. The decision maker may be expert in his own right, in which case the program may justify its existence by improving his productivity.

Expert system technology derives from the search discipline of Artificial Intelligence (AI): a branch of computer science concerned with a design and

implementation of programs which are capable of emulating human cognitive skills such as problem solving, visual perception and language understanding. The typical tasks for expert systems involve:

- The interpretation of data,
- Diagnosis of malfunctions,
- Structural analysis o complex objects,
- Configuration o complex objects,
- Planning sequences of actions.

1.2. THE CHARACTERISTICS OF AN EXPERT SYSTEM

An expert system can be distinguished from a more conventional applications program in that:

- It simulates human reasoning about a problem domain, rather than simulating the domain itself. This distinguishes expert system from more familiar programs that involve mathematical modeling or computer animation.
- It performs reasoning over representations of human knowledge, in addition to doing numerical calculations or data retrieval. The knowledge in the program is normally expressed in some purpose language and kept separate from the code that performs the reasoning.
- It solves problems heuristic or approximate methods, which, unlike algorithmic solutions, are not guaranteed to succeed. A heuristic is essentially a rule of thumb that encodes a piece of knowledge about how to solve problems in some domain. Such methods are approximate I the sense that (i) they do not require perfect data and (ii) the solutions derived by the system may be proposed with varying degrees of certainty.

1.3. DECISION MAKING

Decision making implies the existence of a minimum of the followimg four factors:

1. There must be a problem.
2. There must be a decision maker.
3. There must be alternative solutions to the problem.

Given that these four elements do exist, there are a variety of methods through which one may derive candidate solutions to the problem under consideration – for presentation to the decision maker. The discipline devoted to the

development and implementation of such tools may be called decision analysis. Those who work within this discipline and who ultimately present the alternative solutions to the decision maker, are called decision analysts.

To better understand expert systems, it is vital to understand and a appreciate decision analysis, its supporting elements,its evaloting , and its role in the decision making process. In particular, it is anticipated that through such decision, one may more fully appreciate just when and where to employ expert systems.

Obviously, decision making is hardly new concept. Human beings have been making dcisions ever since human life first appeared on this planet. Cave dwellers had to decide where to live, what to hunt, when to hunt. In making these decisions itis extremely doubtful that they are any rigorous approach to assist them substandiating or improving those decisions made. That decisions were reached strictly by intuition, experience, and judgment.

In more recent times, and in particular and the past few centuries, humans have developed , and have begun to reply on, more formal and rigorous means for assistance in thier decision making. Such means have been achieved through an increased dependence on the use of decision models, and particularly on quantitive models and analytical methods. Today we note their reliance of corporations and institutions on such tecniques as follows:

- Spereadsheets and databases
- Statistical analysis
- Simulation
- Methods of mathematical optimization

While this formal approach to decision making has certainly not subplanted the use of intiution, experience, and judgement, it has found acceptance and use as an adjunct to the decision making process. Typically, when we utilize this more explicit, analytically base approach to decision support, we call it decision analysis to distinquish ıt from the qualitive acpects involved in making decisions. However, ultimately both qualitive and quantitive factors must be taken into account in the decision making process.

The purpose of decision analysis is to provide the deciison maker with information for use in the support of the decision making process, where such information has been derived through a logical and systematic process.

1.4. DP, MIS, & DSS

One way in which decision analysis might reasonably be viewed is a process which involves transformation of the data into (useful) information

support of the decision making process. As our civilations have evolved, we have become great collectors of data. Unfortunately, data alone are of little benefit. To have value, data must be transformed into a format from which we can perceive such useful information trends, measure of central tendency, and measuers of dispersion or variability. One fundamental rule data is that, to be value, data must be in the right form, in the right place, at the right time.

1.4.1 DATA PROCESSING (DP)

The simplest method for the transformation of data is that of data processing, or DP. Typically, the DP approach is used to transform a set of raw data into the following information:

- Statistics
- Pictorial representations

For example, consider a problem in which data have been collected on engine failuresfor the specific tyepe of military aircraft at several different bases. To simplify our decisions, assume that each base has the same number of total aircraft and each flies the same number of missions each month. Twelwe months of data are given in table.The data in table are termed raw data as they simply in the form in which they were originally collected. We may also consider these data to be our engine data failure database.

Month	Base A failures	Base B failures	Base C failures
1	5	3	6
2	1	2	7
3	4	2	5
4	3	1	2
5	7	0	2
6	4	2	3
7	1	2	2
8	7	2	2
9	4	3	3
10	6	1	5
11	6	1	7
12	5	2	7

TABLE : Engine Failure Data

Now, even though our data processing has been eelmentary and incomplete, we should still find it easier to make the following observations:

- The average monthly number of engine at bases A and C are more than twice those of base B.
- A trend in engine failures at base C seems possible. That failures appear to increase in te winter months and decrease in the summer.

Thus, from this simple illustration, we can see the usefulness of even a very rudimentary level of data processing.

1.4.2. MANAGEMENT INFORMATION SYSTEMS (MIS)

The next level sophistication in the processing of data information is called management inforation systems, or MIS. While there is no uniform agreement about the precise definition of MIS, the general intent of the earliest such systems was to provide information directly, and in real time, to the decision makers. And in a format compatiblewith their style and needs fr decision making. İnformation is the bases upon which managers may purpue their duties, specifically the duties of planning, organazing, staffing, and control. MIS certainly existed before the advent of the computer, itis now customaryto think of a MIS as a system that funishes management information by means of a digital computer and connecting information network. The typical MIS concept involves a computer console display at the decision maker's desk.

1.4.3. DECISION SUPPORT SYSTEMS (DSS)

As may be noted from the above discussion, MIS are relatively passive entities. While they remove mush of the drudgery of the data processing and the development of visual aids, and substantially decrease the time required to obtain such information, they still play a limited role in decision making. However, at about the same time that MIS was becoming popular,developments were taking place in other fields that addressed the implementation of certain analytical methods for decision analysis. In particular, representative mathematical models of certain classes of problems were being formulated and various methods for providing solutions to the models were constructed. Including among such methods are following:

- Mathematical programming
- Marginal analysis
- Input-output analysis
- Queuing theory
- Inventory theory
- Project scheduling

- Simulation
- Reliability and quality control
- Forecasting
- Group technology
- Material requirements planning

Asumming that can represent our specific problem using one or more of such models, the associated methodology may then be used to develop a proposed solution. However, as the critics of such approaches have noted , to accomplish this, one is required to transform a real world problem into a mathematical model.

While some advoates of DSSs might disagree one may think of a DSS as a combination of a MIS and the analytical tools as listed above. Thus one conception of a DSS is that computerized system for accessing and processing data, development managerial displays, and providing recommended courses of action as developed through the use modern anlytical methods. Using this definition, a block diagram dor a general DSS is depicted in Figure at below:

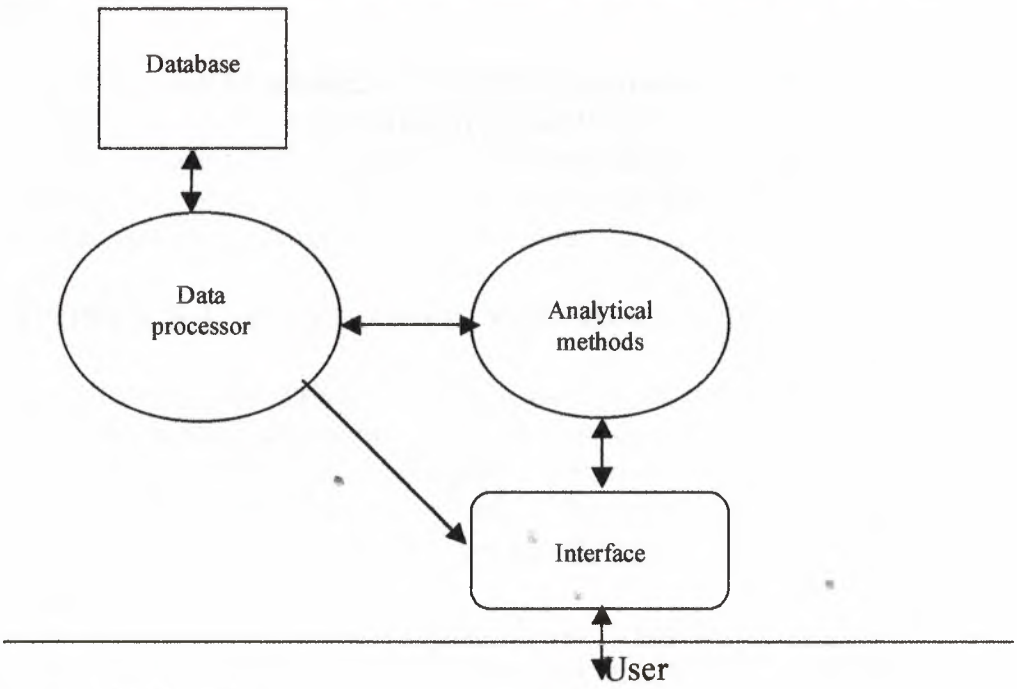


FIGURE – A genetic DDS

As in the case of the MIS, the DSS would access the database and develop displays in the appropriate format. However, assuming that our DSS includes a supportingtool for the solution of scheduling problems, the manager willalso be provided with a recommended schedule for production as generated by the scheduling methodology. The manager may then either accept the DSS

recommendation or develop his or her own schedule – which may be compared with one developed by the DSS through a simulation of the proposed schedule, forexample. Thus, a DSS is certainly a far more active participant in the decision making procdure than either DP or MIS.

Through discussion of DSS, we have referred rather casually to analytical methods. Such methods normally invoke the use of algorithms for the derivation of the solutions for the particular class of mathematical model under consideration. To more fully appreciate the DSS concept, as well as the difference between DSS and experet systems, we need to understand algorithms.

1.5. ALGORITHMS AND RELATIONSHIPS

One formal definition of an algorithm is " a method for solving a problem using operations from given set of baisc operationswhich produces the answer in a finite number of such operations". Typically ,these basic operations are simply elemenatry mathematical procedures such as addition, subtraction, multiplication, and devision. Note most carefully that this definition implies that an algoirthm converges.

Algorithms may be applied to a either single mathematical relationship or (and more likely) to set of such relationships, for the purpose deriving a solution. A mathematical relationship is simply a mathematical stsatement that relates the various components of asystem. In other word, a relationship is a representatin of our knowledge of how a particular systems works.

1.6. HEURISTICS AND HEURISTIC PROGRAMMING

Heuristic rules, or heuristics for short, are that are developed through intituon, experince, and judgement. Typically, they do not represent our knowledge of the design, or interrelationships within, a system. Heuristics do not necessarily result in the best, or optimal, result. Heuristics are often called rules of thumb. For example, consider the following heuristics:

- Don't ask the boss for a raise if he is in bad moon.
- Avoid Houston's Southwest freeway during the rush our.
- Sell a stock if the dividends are tobe cut.
- Buy gold an inflation hedge.

One of the general characteristics of many heuristics is their focus on screening, filtering and pruning. Weach of these terms represents just another way to sate that heuristics may be used to reduce the number alternatives that are

considered. Typically an expert learns through time and experience that certain appoaches tend to work well, while others do not.

When one or more heuristics are combined with a procedure for deriving a solution from these rules, we ahve a heuristic program. Ass in the case of algorithms, heuristic programming involves finding a solution to a probelm using operations from a given set of basic operations, where such a solution is produced in a finite number of such operations.However, and this is the main differance between algoritmic produces and heuristic programming., the solution found may or may not be a thoretically best possible answer.

Not that when one uses heuristics, heuristic programming , one is implicitly accepting the notation satisficing. Satisficing is concept for use in the explanation of how individuals and organizations actually arrive at decisions. Specifically, we typically do not seek optimal solution; rather we seek an acceptable solution. Heuristics (heuristic programs) are then idended for use in obtaining acceptable solutions. However, we can only justify the use of heuristics in those cases for which more formal anlytical methods (in particular, methods that develop optimal solutions) would prove less effective.

At some point, even such mathematically sophisciated approcahes will no longer work. This is , becuse such a problem exhibits has been called combinatorial explosiveness. That is, the time required to solve such problems increases exponentially with problem size. In such instance, we might be well advised to to heuristics and heuristic programmıng simply because of the computational complexity of the problem.

In heuristic programming, we have, in essence, the same situation. That is the heuristic rules comingle with the steps of the solution procedure. However, in this instance, our solution procedure is not algorithm, as it is does not guarantee an optimal solution. On designation for the solution procedure is that of an inference process – the procedure which serves to infer conclusions from the set of the heuristic programming for processing on a machine. The heuristics used to provide a solution (schedule) for this problem involve the following:

- Schedule jobs with shorter processing times before those with longer ones.
- If two (or more) jobs are tied for processing times, give priority to the job that is mostly tardy – or likely to become tardy.

Application of these rules, through a huristic program, will certainly result in a scehdule. Hopefully, such a schedule might even be a good one – but there are no guarantees as to how close, or far, we migth Be from the optimal schedule. In order to keep the discussion simple.

We may does conclude that heuristics and heuristic programming , when and where appropriate, may enhance one's decision making procedure. As such, these methods often form a portion of the tools incorpareted inot decision support systems and serve to alleviate the limitations of the more rigorous analytical techniques.

1.7. ARTIFICIAL INTELLIGENCE

Artificial intelligence, or AI concerned with precisely the same problem that DSS anh heuristic programming are concerned with, that is, decision making. One fundamental differnce is the objective of those in the AI community considerably more ambitious than that of the DSS sector. The purpose of AI is not simply to support decision making, or making to enhance decision making; rather, the ultimate goal of AI is to develop an intelligent machine that will itself make decisions. In particular, this intelligent machine should exhibit intelligence on the same order as that a human.

An intriguing definition of AI is " AI is the study of how the make computers do things at which, at the moment , problem are better". Using this definition , we may avoid the problems of either the definition of determination of the existance of intelligence and instead, simply compare the computer's performance (in some are) with that of humans.

1.8. CERTAIN DIFFERENCES OF OPINION

Much of the criticism now being directed toward expert systems is, we believe, due to rush become involved with the methodology coupled with a failure to take the time and effort to truly understand and appreciate the concept , its history, scope, and limitations. In addition, we must admit to disareement with a number of commonly helds perceptions of , and practices within, the expert systems. The include ,in particular , the following:

- The implication, in the literature, through omission of statements to the contrary , that expert systems can and should be used in virtually any problem. And a failure to emphasize the importance of having a reosanablefamiliarity with the alternative solution procedures.
- The implication that, to understand and use expert systems , you maust be familiar with certain AI languages(LISP and PROLOG)
- Statements that imply that expert systems is just an " alternative and conventional computer programming"
- The emphasis, in too much of the expert systems litearature, on those factors that really only support expert systems.

- The widely held belief that a knowledge engineer is sysnonymous with a computer programmer – or computer scientist.
- The implication that one way learn how to useexpert systems by simply learning how to run a commercial expertsystems software package – and the resulting the development of expert systems software technicians, rather than compentent knowledge engineers.
- The blief that, just because a person doing a job, he or she is an expert in that job – ant the concominant cloning mediocrity.
- The widespread belief that best, if not only to validate the performance of an expert system is to compare its performance.
- The belief that potential expert systems developers should look at applications that have the potential of either saving the company or earning for the company several milion dolars a year. This further imlies that the only expert systems worth building are those incolving many hundreds or tousands of rules.

1.9. APPLICATION OF EXPERT SYSTEMS

1.9.1. DENDRAL – An Expert in Chemical Identification

Work on DENDRAL, generally considered to be the very first expert system. The purpose of DENDRAL, which did not actually become operational until the early 1970s, is the identification of the molecular structure of unknown compounds , a problem of considerable computational complexity. DENDRAL, unlike many of the early expert systems found acceptance and is still in use by chemists all over the world.

The purpse of the collaboration was to determine if heuristics could be used to develop results comparable to the algorithm, but in less time. DENDRAL utilizes production rules and was implemented in the LISP programming language.

1.9.2. HEARSAY I & II – Speech Recognition

HEARSAY I &II were developed in attempt demostrate the possibility of a speech recognition system. Specifically, the goal of the system was to have a computer understand spoken input. The input to the HEARSAY system is a speech waveform. From this waveform, a set of hypotheses about may have been said is developed. A best guess from this set is then presented as the output.

One of the more innoative concepts developed by the HEARSAY project was that of the use of multiple knowledge bases.

At the completion of the HEARSAY project in 1975, the system had a vacabulary of about 1000 words and was able to correctly interpret spoken input roughly 75 percent of the time.

One of the important result of this project was the demonstaration that an expert systems approach was superior to what had been theconventional approach to speech recognition.Included anoung these are HASP/SIAP systems.

1.9.3. INTERNIST/CANDUCEUS – An Expert in Internal Medicine

The INTERNIST project was started in the early 1970s, and continues today under CADUCEUS. One of the truely striking things about INTERNIST/CANDUCEUS has been its ability to remain viable project over such an extensive period time.

The goal of INTERNISTis to perform diagnosis of the majority of dieases assciated with the field of internal medicine.This, in itself , is an ambitious endeavor as there are hundreds of such dieases.

1.9.4. MYCIN – An Expert in Blood Infections

MYCIN is, at this time, probably the most widely known of all expert sysetms. And this despite the fact that it has never been put into actual practice. MYCIN system – and the project has served to substantially infuluencemuch of the sub seqeunt work in the construction and the implementation of expert systems.

The particular role proposed for MYCIN was that of providing assistance to physicians in the diagnosis and treatment of meningitis and bacteria infections. MYCIN is thus somewhat akin to INTERNIST/CADUCEUS inits purpose, except that it focuses on a far smaller number of diseases and thus requires a considerably smaller knowledge base.

The knowledge base of MYCINcontains the heuristic rules. EMYCIN(for empty MYCIN) is the name given to MYCIN when this specific knowledge base is removed.The result of incorporating aknowledge base associated with pulmonary disordersi into EMYCIN resulted in a new expert system known as PUFF.

1.9.5. PUFF – An Expert In Pulmonary Disorders

PUFF was developed using the EMYCIN shell. The purpose of PUFF is to interpret measurements related to respiratory tests and identify pulmonary disorders. PUFF ınterfaces directly with the pulmonary test instruments used in such measurements. At the conclusıon of the testıng, PUFF presents the physivian

with its interpretation of the measurements, a diagnosis of the ilness, and a proposed treatment scheme. The first version of PUFF had 64 production rules. A more recent version had about 400 rules.

1.9.6. XCON(R1) – An Expert in Computer Configuration

XCON (originally tilted R1) was developed for the configuration of VAX computers. A VAX computer may be configured in an enormous number of ways, and it attempts to configure each according to the specific requirementsof each customer. XCON consisted of more than 8000 production rules running under the OPS5 environment (an LISP besed system that typically operates in a forward changing mode).

1.9.7. DELTA/CATS – An Expert in the Maintanence of Diesel-Electric Locomotives

DELTA/CATS-1 consist of knowledge base (i.e., set of heuristic rules) that was acquired through interviews. The system was originally developed in LISP and then converted to forth for increased transportability and speed of execution. Both forward and backward cahaining are utilized.

A particularly interesting feature of DELTA/CATS-1 is its interface with visual support systems. More recently, rumors have circulated that DELTA/CATS-1is having a problem similar to those cited for XCON, and the system may, in fact, have been shelved.

1.9.8. GATES – An Airline Gate Assignment and Tracking Expert System

GATES is in used evidently in prototype form . The system is being used to assist ground controllers in the assignment of gates to arriving and earting flights. The knowledge base was acquired from an experienced ground controller who solved such problems on a daily basis.

The gate assignment problem can become quite complex, and reqires rapid solution during intervalsof flight delays, bad weather, mechanical failures, and so forth.Gates was developed, using the PROLOG, implemented on a personal computer.

1.9.9. QMR – Medical Diagnostic Expert System

Using the knowledge base first developed for INTERNIST, QMR assistsphysicians in the diagnosis of an illness based upon the patint's

sysmptoms,examination findings, and laboratory tests. QMR, incorporatesover 400 possible manisfestations of dieseases and is said to perform at level comparable to practising phycisians.

1.9.10. FXAA – Foreign Exchange Auditing Assistant

This involve thousands of transactions a day with a paaprework resulting from such transactions wighing i at about 10 pounds per month. FXAA has been developed to provide the ecessary auditing assistance.FXAA is a rule- based expert system that has evidently made a major impact within Chemical Bank.

1.9.11. Jonathan's Wave – An Expert Commoduties Trading

A number of firms and individuals have developed expert systems for stock and commodity trading. While itis still too early to asses the success of failureof these programs. Jonathan's Wave runs on two 286 based personal computers. The knowledge base is written in C while the inference engine is written in PROLOG.The sysem acts somewhat as through it were using multiple experts to reach its conclusion.

1.9.12. Insurance ExperTax – An Expert in Tax Planing

Coopers and Lybrand have created Insurance ExpertTax to assist in the identification of tax planning and accrual issues.Insurance ExperTax took more than a year to develop an dconsists of more than 3000 rules. Created in LISP amd running on the IBM PC.

1.9.13. HESS – An Expert Scheduler for the Petrochemical Industry

HESS was developed in support of product scheduling at a mojor petrochemical firm's refinery. The knowledge base in HESS was developed via the acquisition of heuristic rules from two refinery product schedullers. HESS was developed using the EXSYS expert system shel, through a 12-month effort. HESS, which stands for hybrid expert system.

1.9.14. An Expert Poultry Farming

They is developing an expert system for the poultry farmer. The system utilizes the Nexpert Object expert system shell. The system analyzes data from the poultry farm's environmental control system. Using information on feed and water consumption, temperature, humidity, and ammonia levels, the system may be used to alert farmer to any diseases the chickensahve, or may get.

1.9.15. Dustpro – An Expert in Mine Safety

Using the Level 5 expert system shell as a development vehicle, it has developed an expert system named Dustpro. Dustpro replaces the limited number of human experts that assess the air quality of mining operations. Based on the amount of coal and silica dust in the air, mining operations must be adjusted to ensure that safety requirements are satisfied.

1.9.16. TOP SECRET – An Expert in Security Classifications

Within the Department of Energy (DOE), there are more than100 classification guides to nuclear weapon security data. One of the more onerous tasks within the DOE is to attempt to correctly classify a given document through the use of these guides. Document classification determines who is permitted to view a document, and who is not – a potentially critical factor in national security.

1.9.17. Codecheck – An Expert in Computer Assessment

An expert system for evaluation of C codes, termed Codecheck, the package is a rule-based expert system that checks C source code for such things as complexity, formatting, and adherence to standards.

The most common cause of hard to maintain software is the programmers' tendency to write overly complex code. Codecheck identifies those portions of the code may be simplified. In addition, Codecheck evaluates the portability of the source code by the comparing it with the numerous standards now existing for C programs.

1.9.18. Expert Systems for Faster, Fast Food Operations

Expert systems have even permeated the fast food market. A recent article describes the introduction of expert systems into such companies as McDonalds … Here, such systems serve to reduce inventory, speed up service, and even act as training assistants. Packages provide valuable, timely assistance to managers who are neither familiar, not entirely comfortable with the pace of activities in such operations. In sector in which there exists such fierce competition, any improvement in cost reduction and enhanced operations can simply not afford to be overlooked.

1.10 AN EVALUATION OF PROBLEM TYPES

Although just a few examples of expert systems has presented, we might note that they are fairly representative of the bulk of applications thus far

developed. That is, the majority of applications involve classification (diagnosis). For example, in the medical expert systems, we are given certain data (symptoms) with regard to a patient and attempt to diagnose the associated cause, disease. In maintenance applications, precisely the same type of problem is faced. Here, the symptoms are the data on machinery performance while the diagnosis involves the identification of a defective or failed component. Further, once a classification has been made, the specific class is matched to an associated treatment.

The remaining set of applications involves what is defined in this text as construction problems. XCON and HESS are representative of this type of application. Note that XCON attempts to construct a VAX computer, while HESS attempts to construct a schedule.

1.10.1. Classification & Construction Problems: Definitions

Classification as an attempt to draw boundaries about existing elements. For example, a certain set of existing symptoms point to a particular disease. Construction, on the hand, seeks to determine the arrangement of elements. That is, classification problems usually require backward search (backward chaining) while construction problems typically require forward search (forward chaining).

Another, more visual, means for discriminating between these two fundamental types of problems is available by means of nothing just how each type of mental is mapped. To illustrate, consider figure below. On the left of this figure, we have 5 objects. Associated with each value of these objects are certain attributes and values. On the right side of the figure, we have mapped these 5 objects into two groups.

To further clarify this concept, consider a problem in which the 5 objects on figure below are five different automotive engine parts. Each object is a set of data pertaining to various quality tests. Further, we simply wish to distinguish between parts that are acceptable and those that are not. Thus, a priori, we have two classes. Using the data set, a quality control engineer may then assign each object to one of the two classes. And is a typical classification problem.

Next, let us assume that the problem involves the loading of 5 items onto a fleet of trucks. Initially, we are not sure how many trucks are necessary. Associated with each item such attributes as weight, volume, cost, and priority. Using the values of these attributes, the cargo loader (or expert system) will then determine the loading scheme

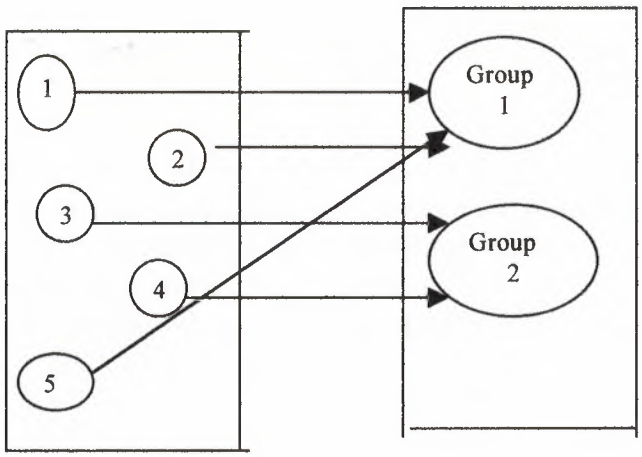


FIGURE – Mapping the objects to grouping

Thus, above figure depicts a scheme in which items 1, 2, and 5 are loaded on one truck while items 3 and 4 are loaded on another truck. Again, note that the determination of the number of trucks used is an integral part of this problem, which is representative of a typical problem of construction.

A more recent, and more precisely defined, attempt toward problem classification has been defined accomplished. He lists four types of applications for AI (or expert system):

- Class I: Characterized by a need to select a solution from a fairly well - defined set of possible alternatives – such as the medical diagnosis problem. This class coincides to what we have termed as classification problems.
- Class II: Characterized by a need to create a plan or configuration and scheduling. This class coincides to what we have termed as construction problems.
- Class III: Characterized by need for the true creativity. Such problems include those of design, including those where the very nature of the problem itself might have to be redefined.
- Class IV: Characterized as applications that humans can handle and computers can't. Included among this class are such problems as face recognition, reasoning by analogy, and learning how to talk.

1.11. FUTURE EXPERT SYSTEMS

Most of the expert systems that have thus far been discussed in the literature are essentially stand-alone systems. However, in the very near future it is likely that a large portion of the expert systems that are developed will be

embedded systems, that is, systems that form only part of the overall software package.

Another form of the embedded expert system is that of the so-called intelligent interface. Intelligent interfaces shall rely, more and more; on expert systems to be better achieve user friendliness in software. Such a system will immediately determine whether or not the user is a novice or expert, and its tailor actions accordingly. That is, the novice user will require more help, support, and guidance, while the more experienced user will need but minimal assistance.

Another trend that we expect to continue is the increased development expert systems – expert systems having 200 or fewer rules. This particular prediction is, however, somewhat, at odds with a commonly held belief of the AI community. A view may have made sense some years ago when expert systems development was somewhat of a trial and error process, and when much of the software for support had to be developed by one. And in a difficult language such as LISP, and with the support of expensive LISP machines. However, with the advent of powerful, inexpensive expert shells – and with implementation on the personal computer – the development of small expert systems are highly cost effective.

CHAPTER 2

ARCHITECTURE OF EXPERT SYSTEM

2.1 THE STRUCTURE OF EXPERT SYSTEM

Figure 1 depicts one possible representation of an expert system. The Components above the dashed line are those with in the computer. Below this line, access capabilities for two types of human users are noted. The first is that individual designated as the knowledge engineer. As discussed, the knowledge engineer is the person responsible for placing into the expert system's knowledge base; the portion of the expert system shown figure 1below. He or she accomplishes this through the *interface* and *rule adjuster*.

The knowledge engineer is also interface between human expert and the expert system. That is, the knowledge engineer somehow must capture the expertise of the human expert and then express this expertise in a format that may be stored in the knowledge base-and will be used by the expert system. In the ideal expert system, there would be no need for a knowledge engineer. The domain expert would interact directly with the expert system and would replace knowledge engineer in the figure.

The second type of individual with access to the expert system is designated, in figure 1 , as simply the user. This designation refers to anyone who will be using the expert system as a decision making aid. And the successful knowledge engineer must always keep in the mind that the expert system is ultimately intended for the benefit of the user, not for that of the knowledge engineer or the domain expert.

The interface handles all input to the computer and controls and formats all output. The interface would handle such scores. A well-designed interface would be one that exhibits ease of use, even for the novice user. The interface also handle all communication with the knowledge engineer during the development of expert system's knowledge base. Another property that sometimes exhibited in expert systems is that of *explanation.* That is, some expert system have limited ability to explain the reasons for the any questions asked of the user, as well as the rationale for the conclusion reached. Again this function would be the responsibility of the interface.

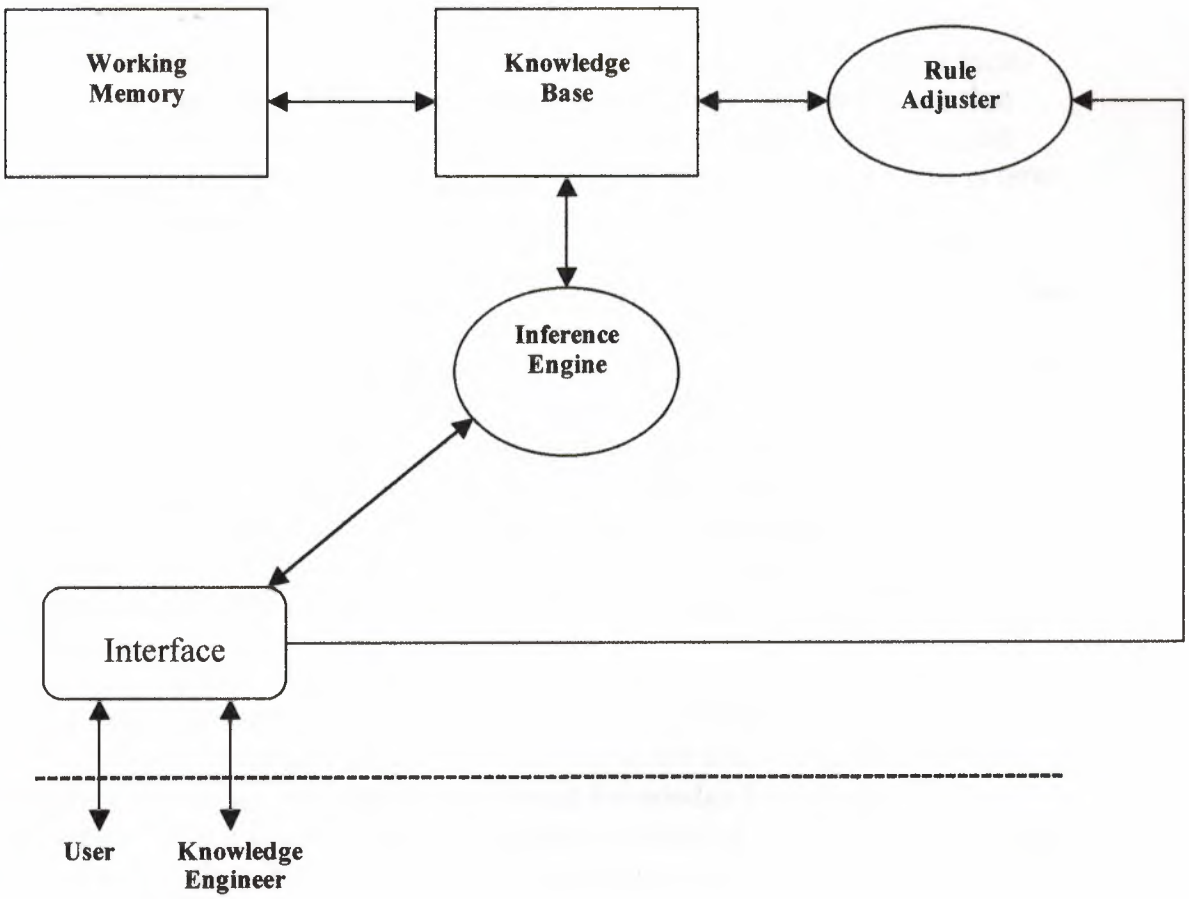


Figure 1 – A generic expert system.

The interface engine is employed during a consultation session. During consultation, it performs two primary tasks. First, it examines the status of the knowledge base and working memory so as to determine what facts are known at any given time, and what facts are known at any time, and to add any new facts that become available. Second, it provides for the control of the session by determining the order in which inferences are made. An alternative designation for the inference engine, and perhaps a more element appropriate one, is that of *knowledge processor*. As the knowledge-processing element of an expert system, the inference engine serves to merge facts with rules to develop, or infer, new facts.

The knowledge base is, as we have emphasized repeatedly, the very heart of any expert system. A knowledge base will typically contain two types of knowledge, that is, facts and rules. The facts within a knowledge base represent various aspects of a specific domain that are known prior to the exercise the expert system.

The Working memory of an expert system changes according to the specific problem at hand. The contents of the working memory consist of facts. However unlike the facts within the knowledge base, these facts are those that have been determined for the specific problem under consideration during the consultation session. More specifically, the result of the inference process is new facts and these facts are stored in the working memory.

The final module discussed in the rule adjuster. In most expert systems, this serves merely as a rule editor. That is, it enters the rules specified by the knowledge engineer into the knowledge base during the development phase of the expert system. It may also allow for various checks on these rules. In more ambitious expert systems, the rule adjuster may be used in an attempt to incorporate learning into the process. In such instances, teach expert system by providing it with a set of examples and then critique its performance. If its performance is unsatisfactory, the rule adjuster automatically revises the knowledge base. If satisfactory, the rule adjuster may simply reinforce the existing knowledge base.

An expert systems "shell" includes all of the components listed figure 1 minus the knowledge base. Using a shell, it is up to the knowledge engineer to develop the knowledge base and to then insert knowledge base into the architecture to form a complete expert system, as intended for a specific domain. The use of a shell thus frees the knowledge engineer from the need repeatedly develop all supporting elements of a expert system, and thus the focus his or her attention on the development of the knowledge base.

The architecture of generic expert system, as depicted in figure1, should serve to indicate at least some of differences between this approach and that of algorithmic procedures and heuristic programming. In particular note that the knowledge base is separated from the inference engine. In other words, and unlike algorithms and heuristic programming, an expert system separates heuristic rules from the solution procedure. The knowledge base contains a description, or model, of "what we know". The inference engine contains a description of "what we do " to actually develop the solution. While the knowledge base changes from domain to domain, the inference engine remains the same.

2.2 ON THE ORIGIN OF EXPERT SYSTEMS

When one examines the bulk of the artificial intelligence or expert systems literature, one is given the definite impression that expert systems is a relatively new concept and is a concept that owes its existence primarily to the so called AI community. There are others (e.g. individuals in the fields of management science,

operations research, and in particular) who take strong exception to this perspective. Rather than viewing expert systems as an entirely new idea, they would consider expert systems simply another format for the implementation of heuristic programming. Under this particular view, expert system is then neither new nor a product of either AI or computer science.

However, while heuristics and heuristic programming have been in use for centuries, these approaches have rarely been considered *scientifically acceptable*. A paper describing a heuristic programming approach to some problem would, until fairly recently, have had little change for publication in most professional journals. In such fields as operations research, heuristic programming was felt to lack rigor and credibility of algorithmic approaches – *even, oddly enough, when the approach solved problems that were far beyond the capabilities of the analytical techniques*. Expert systems, however, have receiver rather obvious, and quite prominent acceptance. This acceptance is due, in large measure, to various contributions by the AI community coupled with highly successful promotion of expert system approach. Heuristic programming has generally been an unstructured field of endeavor, with little in the way of a general set of guidelines, rules and produces. Further, those who practiced within this field were all too often looked upon with the some degree of contempt by their more mathematically gifted contemporaries. This is in considerable contrast to the treatment of heuristics (as expert system) by the AI community.

The AI community, while using heuristics as the essential ingredient of the expert systems knowledge base, has augmented its use with very formal set of policies and procedures. That is, some much needed discipline and rigor to the heuristic programming approach have been added? Further, the importance and credibility of the heuristic (or expert system) approach have been recognized and efforts in the area have been encouraged. This, while heuristic programs and expert systems share much in common, including the fact that both suffer from the same limitations, it is expert systems that has proven be both commercially and scientifically acceptable.

The primary enhancements brought to heuristic programming from the AI community are not trivial. For example, the separation of the inference process from the knowledge process is a concept, which at first glance might seem minor. However, do not misled. This separation has resulted in the ability to focus our efforts much more intensely toward the development of the knowledge base model – rather than have spend an inordinate (and generally unnecessary) amount of time and resources in the development and implementation of the entire solution process. Since it is the model that determines the outcome, whether one uses algorithms, heuristic programming, or expert systems, it should be apparent that this is the area to which we should devote the bulk of our efforts.

CHAPTER 3

KNOWLEDGE REPRESENTATION

3.1 COMPONENTS OF KNOWLEDGE IN EXPERT SYSTEMS

The knowledge that is contained within the an expert system consists of:

- *A priori knowledge:* the facts and rules that are known about a specific domain prior to any consultation session with the expert system.
- *Inferred knowledge:* the facts and rules concerning a specific case that are derived during, and at the conclusion of, a consultation with the expert system.

The major goal is to represent the facts and rules within the knowledge base of expert system . For this reason, it is needed to

- Provide a format compatible with the compute.
- Maintain as close as possible a correspondence between this formats and actual facts and rules.
- Establish a representation that can be easily addressed, retrieved, modified, and updated.

Elaborating further on the last two points, it would be highly desirable to use a format that is *transparent*, that is, a representation scheme that may be easily read and understood by humans.

Several modes of knowledge representation have been proposed. The primary focus will be on rule-based systems for knowledge representation, since it is through this process that the knowledge bases of the expert systems to be described in this text will be developed.

3.2. ALTERNATIVE MODES OF REPRESENTATION

There are following modes of knowledge representation : Object attribute value (OAV) triplets, semantic networks, logic programming, frames , neural network production rules.

3.2.1. OAV Triplets

Object – attribute –value triplets provide a particularly convenient way in which to represent certain facts within a knowledge base and may be extended (as we shall) to provide basis for the representation of heuristic rules. Each OAV triplet is concerned with some specific entity, or object. For example, our object interest might be an airplane. Associated with every object is a set of attributes that serve to characterize that object. Using the airplane as an example (i.e., as the object), some of its attributes include the following:

- Number of engines
- Type of engine (e.g., jet or prop)
- Type of wing design (e.g., conventional or swept back)

For each attribute, there an associated value, or set of values. For instance, in the case of the C130 military cargo aircraft (known as the Hercules), the number of engines is four, the type of the engine is prop, the wing design is conventional. Notice in particular that values in OAV triplets may be numeric or symbolic. We may list these facts as shown below:

- Number of engines = 4
- Engine type = prop
- Wing design = conventional

Observe that, in this list, the object itself (i.e., the C130 aircraft) is never explicitly stated. Actually, the above statements represent AV (attribute-value) pairs. However, associated with any AV pair is some object. Thus, any AV pair implies an OAV triplet.

Yet another way to represent an OAV triplet would be through the use of a network representation as indicated in figure below. The basic building blocks of a network are its nodes (i.e., circles) and branches, or edges, (i.e., the lines connecting two nodes). In figure below, the object is Pete Jones, the attribute is his income, and the specific value of his income is $50.000.

FIGURE- OAV network.

3.2.2. Neural Networks

The human brain stores knowledge. NN represent mankind's attempt to replicate, in hardware, theories pertaining to the brain. Specifically, it is thought that knowledge is stored in neurons (or in the connections between neurons). Figure-1 depicts simplified representation of only two neurons within the NN of the human brain. In the human brain there are more than 10 billion neurons and each neuron is connected to one or more neurons, resulting in a massively interconnected network. At each neuron, impulses are received by the dendrites and transmitted by the axons. Each neuron has synaptic connection weight. Knowledge might be represented by the weights of each neurons interconnection, which in turn influence the level of strength of the interconnecting impulses. To duplicate the NN structure the software and hardware tools are used. These are number of programs which simulate NN. Typically, electronic amplifiers, chips etc. are used to represent neurons. The resistors are used for interconnection. By NN are used for speech recognition, pattern recognition, classification, control etc.

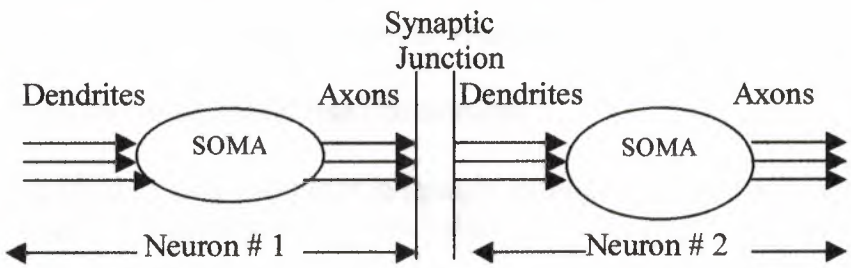


FIGURE – A portion of neural network.

3.2.3. Representation via Logic Statements

The most common form of logic is that known as propositional logic. A proposition, in turn, is a statement that may be either true or false. Propositions may be liked together with various operators (termed logical connectives) such as AND, OR, NOT, and EQUIVALENT. Linked propositions are termed compound statements. To demonstrate, consider X, Y, and Z, where first two statements (X and Y) are true while Z is false. Thus, we may conclude that:

- X AND Y is true and X and Z is false. If two statements connected by AND, both must be true for compound statement to be true.
- X OR Y is true and X OR Z is also true. If two statements are connected by OR, that statement is true as long as either one or both statements are true.
- NOT Z is true. The NOT connective simply negates the statement. Since Z was false, NOT Z must be true.

Predicate calculus represents are extension of propositional logic. The fundamental elements of predicate calculus are the object and the predicate. A predicate is a simply a statement about the object, or relationship that the object processes. Predicate may address more than one object and be combined by use of logical connectives. For example,

- Mammal (dog), which is read as " a dog 's a mammal "
- Four-logs (dog), which is read as " a dog has four logs "
- Mammal (chicken), which is read as " a chicken a mammal "
- Sister (Joan, Jan), which is read as " Joan 's Jan's sister "

The 1-st statement of is true, the 2-nd is in general true, and is 3-rd is definitely false. Unless we know Joan and Jan personally, the validity of the fourth statement is unknown. Using the predicate calculus, we can than represent such compound statements as Joan is Jan's sister Fred's cousin" as sister (Joan, Jan) and cousin (Joan, Fred).

We can also represent various relationships or rules by means of predicate calculus.

Fall (bond-prices) if rise (interest-rate).

Here, we used Prolog computer language. Prolog is logic programming language and is used for representation knowledge.

3.2.4. Semantic Networks

A semantic network may be thought of as network that is composed of multiple OAV triples in the network form as illustrated in figure-1. Semantic networks may be used to represent several objects, and several attributes per object. We might develop a partial semantic network as illustrated figure-2. Here, we note that the CSA is a special type of aircraft (i.e. a large military cargo plane). Further since the CSA is an aircraft in general, it inherits the properties associated with the aircraft in general (e.g. it flies, has wings, carries people). Such an inheritance property can prove to be of considerable value in the reduction of memory storage requirements. That is since a CSA is an airplane, there is no need to store, at the CSA node, the fact that it can fly, has wings, and can carry people. Thus, the semantic network scheme provides for a convenient approach for the representation of associations between entities.

We might also note that the OAV triplet is actually first a restricted subset of semantic networks wherein only relationships that may be used are those of " is–a" and "has-a". OAV nodes, in turn may be any three types: objects, attributes or values.

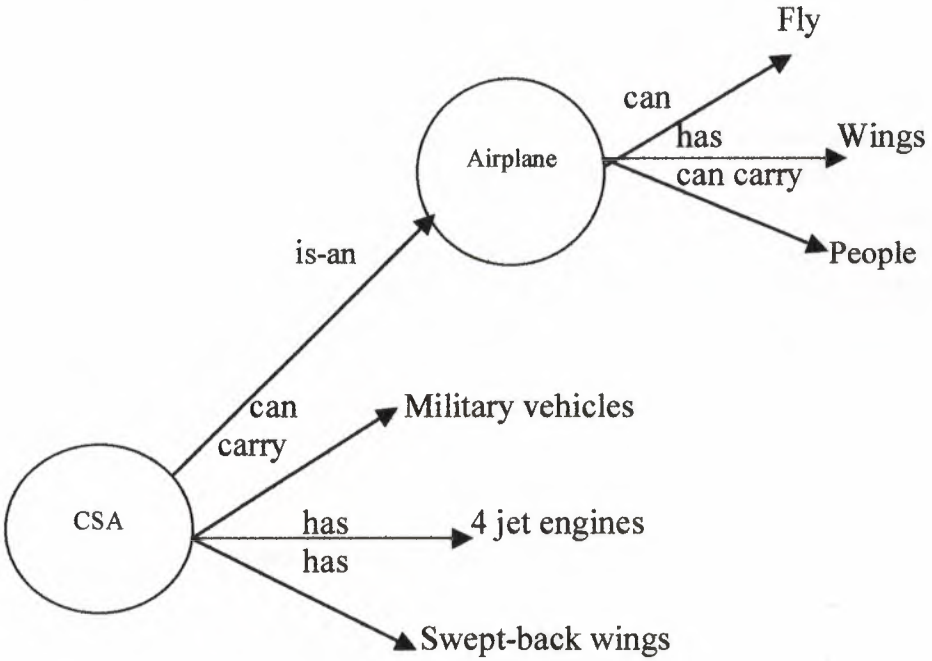


FIGURE – Semantic network

3.2.5. Frames

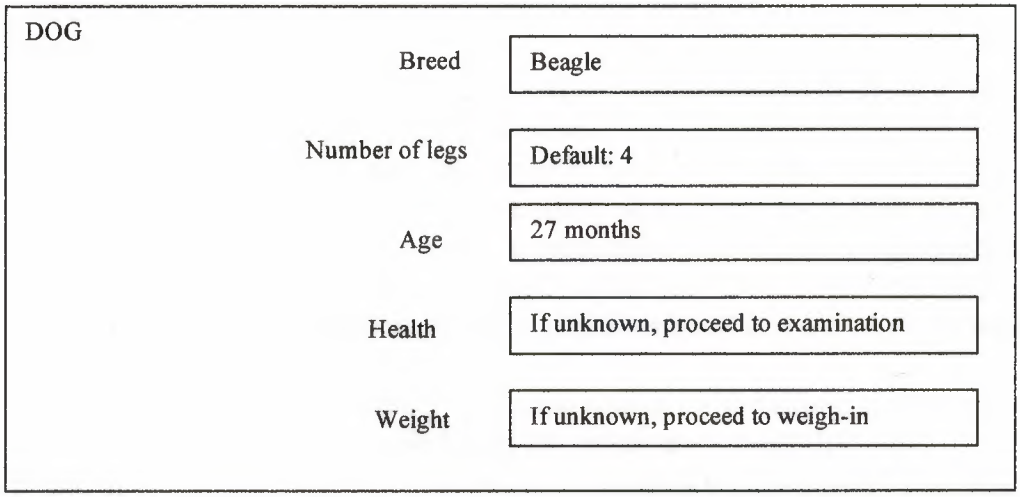
While semantic networks provide relatively versatile means for knowledge representation, the use of frames represents an alternative approach that serves to capture most of the features of the semantic network while providing certain additional aspects. In fact, we may think of a semantic network as being a subset of the concept of frames.

The employment of frames represents a particularly robust way in which to present knowledge. A frame contains an object plus slots for any and all information related to the object. The contents of such slots are typically the attributes and attribute values, of the particular object. However, in addition to storing values for each attribute, slots may contain default values, pointers, other frames, and sets of rules or procedures that may be implemented.

Figure-3 illustrates a frame-based representation or the object dog. Note that the slots within this frame include values (e.g., Beagle), defaults (e.g. four legs), and procedures (e.g., for a medical examination). The procedures, in turn,

could well point to other frames. The versatility of he frame based mode of knowledge representation should be obvious.

The primary drawback to the use of frames is, caused by the very robustness of such mode of representation. Frames have so many capabilities as to make their use a rather complex matter. A result to obtain any reasonable proficiency in the use of frame-based tools in expert systems, a lengthy training period is required. Despite such drawbacks, frames can prove quite useful, if not essential, in the design of large-scale, complex expert systems – particularly those involving a large amount of a priori facts (i.e., data) and multiple objects.



3.2.6. REPRESENTATION VIA RULE – BASED SYSTEMS

Undoubtedly, the most popular mode of knowledge representation within expert systems, at lest at this time, is the mode obtained through the use of rules, or rule-based systems. Alternatively, such rules are referred to as IF-THEN, or production rules. We have selected rule-based expert systems as our approach to knowledge representation for a number of reasons, including this popularity and widespread use. However, it should be stressed that this decision does not imply that rule-based systems are necessarily the best approach or, in particular, the best approach for every situation. There are those who present quite persuasive arguments for other approaches. Rule-based knowledge representation has been made for the following reasons:

- The majority of existing expert systems development packages employ rule basis.

- Rule-based expert systems development packages are normally much less expensive (in terms of both the initial cost of the package as well as the overall cost of using the package) than those employing alternative modes of representation. Specifically, they cost less to purchase, normally do not require any expensive hardware (most run on inexpensive, general purpose personal computers), and require minimal expenditures toward training.
- The widespread availability of rule-based expert systems shells permits the knowledge engineer to focus his or her attention o the most critical phase of the development of an expert system, that is, on the knowledge base.
- Rules represent particularly natural mode of knowledge representation. Consequently, the time required to learn how to develop rule bases is minimized.
- The learning curve for rule-based expert systems is much steeper than for any alternative mode of representation.
- Rules are transparent, and are certainly far more transparent than the modes of knowledge representation employed by rule-based systems two major competitors: frames and neural networks. Further, such transparency often leads to an increased willingness, on the part of management, to accept the solutions obtained. And the importance of this last factor should be underestimated.
- Rule bases can be relatively easily modified. In particular, additions, deletions, and revisions to rule bases are relatively straightforward processes. And this is particularly so in the case of well designed rule bases.
- Rule – based expert systems can be employed to mimic most features of frame – based representation scheme.
- Validation of the content of rule-based systems is relatively simple process. Similar validation of frames or neural networks, on the other hand, is normally difficult to impossible.

3.2.7. PRODUCTION RULES: AN OVERVIEW

Rule-based modes of knowledge representation employ what are termed production rules or, for short, simply rules. Such rules are typically of the IF-THEN variety. However, in some instances this is extended to include IF-THEN-ELSE rules. For example, we might have the IF-THEN-ELSE rule as shown below:

Rule 1: If the student's score GRE score is 1350 or more
Then admit the student to the graduate program
Else, do not admit the student

Which is equivalent to two IF-THEN rules or,

Rule1a: If the student's score GRE score is 1350 or more
Then admit the student to the graduate program

Rule 1b: If the student's score GRE score is less than1350
Then do not admit the student

For the clarity of presentation, we shall focus primarily on just IF-THEN rules. In fact, it is generally advisable to avoid the use of ELSE statements in rule-based expert systems . This is true for three reasons. First, a number of commercial expert systems development packages simply do not permit the use of the IF-THEN –ELSE rules. Second, validation of such rules is considerably more difficult than for their IF-THEN equivalents. Third, when encountered in the inference process, such rules will tend to always reach conclusion . This can result in some unanticipated results. Thus, whenever one comes upon such a rule , we strongly advise the formation of the corresponding two equivalent rules.

An alternative designation for IF-THEN rules is that of *condition-action* or *premise-conclusion* statements. We shall refer to the IF statement as the premise and to the THEN statement as the conclusion.

We should also realize that there may be several premise and conclusion statements within a single rule. Each of these are termed clauses (i.e. premise clauses and conclusion clauses). Another rule with multiple clauses is in the IF and THEN portions.

Further, not that while premise clauses may be connected by a AND as well as OR operators, the conclusion clauses may only be connected by AND statements. That is, all of the conclusion clauses in a production rule must be true.

Clauses connected by AND operators are denoted as conjunctive clauses. Those connected by OR operators are termed disjunctive clauses.

3.2.8. ATTRIBUTE – VALUE PAIR PROPERTIES

As noted, each premise and conclusion clause contains attributes and values. Further, there must be an associated object, either implied or explicit. Consider, the rule shown below:

Rule 1: if grade point average (GPA) equals or exceeds 3.5
Then accept into honor society

When clauses contain only attributes and values, as in the case of the rule under discussion, they are sometimes called attribute – value or AV pairs. In the conclusion clause, the attribute-value pair is accept into honor society. Actually, this is a poor choice of wording for this conclusion. In general, rules should be written so that identification of the attribute and value is straightforward – while the rules remain intelligible.

The AV pair is the fundamental building block of a premise or conclusion. And thus the fundamental building block of a production rule. Associated with each AV pair is a set of properties. The most typical of these are below:

- **NAME:** The name of the attribute is simply the wording selected to identify the attribute of the object associated with the clause under the consideration. For example, some of the attributes typical of the automobile are color, number of doors…
- **TYPE:** Attribute values may be either numeric or symbolic. For example, the temperature of a patient may be given in degrees Fahrenheit- a numeric value. Alternatively, we might specify the temperature values to be symbolic, such as high or normal. Yet another symbolic values would be yes or no, for example, such as with respect to the presence or absence of some feature.
- **PROMPT:** Associated with certain attributes are user prompts, or queries. When necessary, the user replies to this prompt with a value for the attribute under consideration. The only attributes that should normally be provided with prompts are

  i. Attributes that appear in a premise statement and never appear in any conclusion statement of the rule set
  ii. Attributes for which the user can conceivably provide a response

- **LEGAL VALUES:** Associated with every attribute is a set of legal or acceptable, values. For example, the legal for the person's weight would simply be the set of nonnegative real numbers. If the replies with a nonlegal value, this is detected and the user may be asked to reply again. In the case of expert systems that provide menu-driven prompts the set of legal values is simply presented to the user and he or she can only select from that list.
- **SPECIFIED VALUES:** indicate the actual set of values that are either to be tested against(in a premise clause) or that will be, or have been, assigned(in a conclusion clause). More specifically, we are concerned with whether or multiple specifications are permitted. Multiple values may also be allowed (where, again, this is dependent upon the particular software package employed) for attributes that appear in conclusion

clauses. In other words, it may be permitted to assign (i.e. conclude) multiple values to the attribute in a conclusion clause.

- **CONFIDENCE FACTORS:** If the expert systems development package permits, we may deal with uncertainty in either conclusions (i.e. the conclusion attribute values assigned) or premises (i.e. the premise attribute values used). Since we shall not deal with uncertainty and confidence factors in this chapter, we shall merely note that this too is an AV pair property.

3.2.9. CLAUSE PROPERTIES

As we discussed, there are two types of clauses: premise and conclusion. Other properties associated with clauses are in the below list:

- Single versus multiple (or compound) clauses
- Conjunctive versus disjunctive (multiple) clauses
- Free (premise) clauses
- Specified (premise) clauses (i.e. Specified true or Specified false)

Let us examine each of these properties in turn. First, a premise or conclusion may consist of a single clause or set of clauses.

Multiple clauses. In tern may be either conjunctive clauses (each clause connected by the AND operator) or disjunctive (each clause connected by the OR operator)/ However, recall that disjunctive clauses are not permitted in the conclusion of a rule. Also, note the premise of a rule may be quite complex.

Another property of a clause is that associated with premise clauses only. This is the property of being either free of specified, and if specified, of being true or false. If the value of premise clause attribute is not yet known, that clause is designated as a free clause. Note most carefully that we have drawn a distinction between not yet known and unknown. If a clause is not free, then such a clause is either true or false. Consider the following simple premise clause shown below:

$$\text{If } A = X$$

Note must be attribute for the object which the clause is concerned. X is then one possible (legal) value for this attribute, we must test this clause to see if A does indeed equal X. If we do not know the value for A, and have yet to seek this value, the clause is free. However, if we do know the value for A, and

this value is indeed X, then clause is true. Otherwise, (i.e. if the value of A is known but is something other than X), the clause is false.

The properties of free, true, or false would seem be straightforward. And indeed they are; however a certain degree of confusion may occur when one employs unknown an attribute value. Note carefully that we must differentiate between known and not known yet. Not known yet means that the value for a respective attribute has not yet been determined. Thus the associated clause is free.

Unknown, however, can be employed in one of two ways.

i. It may simply be a legal value for a given attribute. The premise clause is true, and the rule is triggered.

ii. Unknown may be employed is slightly more complex, and a function of the specific mode of the inference used by the software package. In this case a value of unknown is assigned to an attribute whenever its value can not b determined from the inference procedure.

3.2.10. RULE PROPERTIES

As with AV pairs and clauses, there are certain important rule properties. Some of the more typical rule properties are below:

- **NAME:** Each rule should have a distinct, as well as descriptive name. Actually, we have not always followed this guideline. However, it is a good idea to do this when building any actual knowledge base. Specifically, rather than just labeling a rule by a number or letter.
- **PREMISE:** Every rule consists of one or more premise clauses is termed the rule premise. A rule premise may consist of conjunctive or disjunctive clauses.
- **INTERMEDIATE CONCLUSIONS AND CONCLUSIONS:** Every rule consists of one or more conclusion clauses. In the case of multiple conclusion clauses , the clauses must be conjunctive . There are two types of rule conclusions: intermediate conclusions and (final) conclusions. An intermediate conclusion is one that does not appear as a premise clause for any other rule.
- **NOTES & REFERENCES:** It is essential that a rule base be documented. While you, the developer, may know the reason and source of the rules, others will not. Further, with the passage of time, even the developer will find it difficult to recall the origin and specifies of each rule. Many development packages permit the inclusion of notes and references, add this

is a feature that should most definitely be employed in any actual knowledge-base development.

- **(RULE) CONFIDENCE FACTORS:** When uncertainty 's employed, we may associate confidence factors with each rule. Here, we may simply note that the confidence factor of a rule's conclusion is a function of the confidence factors of the rule and the rule premise.
- **PRIORITY & COST:** In some development packages, we are permitted to assign a priority and/or cost to each rule. Such properties are normally employed as a means to decide, during the inference procedure, this specific rule to be dealt with at a particular instance. Typically, the procedure will select the rule with the highest priority or the lowest cost.
- **CHAINING PREFERENCES:** The inference process involves a search procedure. In some cases, the search moves forward direction-from premises (or facts) to conclusions. In other, the search moves backward-from a hypothesized conclusion to the premises necessary to infer that conclusion. However, in addition to such normal modes of search, or chaining, some development packages permits the employment of a mixture of search methods.
- **RULE STATUS:** During consultation, the status of each clause and rule's subject to change. Keeping track of such changes is an essential part of the inference process. We need to become acquainted with the terminology used. A summary of this terminology 's provided below:

  i. The premise of a rule is true whenever a test has been made and it has been determined that the premise has been satisfied.
  ii. The premise of a rule is false whenever a test has been made and it has been determined that the premise has not been satisfied.
  iii. If the premise of a rule is true then that rule is said to be triggered.
  iv. If the premise of a rule is false then that rule may be discarded or, in some cases, made inactive.
  v. If a rule is fired then this implies that action implied by the conclusion clause(s) is taken. The values associated with each attribute of the conclusion clauses for this rule are said to be assigned.
  vi. A rule that has been fired is no longer active. It is either discarded or, in some cases, made inactive.
  vii. If a rule is to be fired, that rule must be first have been triggered.
  viii. If a rule has been neither fired nor discarded, that rule's designated as being active.

3.2.11. RULE CONVERSION: DISJUNCTIVE CLAUSES

While such conversions are not necessary in the general methodology of expert systems, they often make easier for the beginner to follow the inference process of an expert system when manual demonstrations are employed. Further some expert systems development packages do not permit the use of disjunctive premise clauses. However, such a conversion does result in an enlargement of the number of rules necessary to represent the knowledge base of an expert system. Despite this the beginner may be well advised to consider such a conversion – as well as determine the restrictions of the software that is to be used.

3.2.12. MULTIPLE CONCLUSIONS

We must stress, however that it may be quite reasonable for an expert system to draw multiple conclusions – and this is particularly so if we are dealing with the uncertainty.

There are also instances in which multiple conclusions may make sense even though uncertainty is not being employed. To illustrate, consider the three (deterministic) rules listed below:

Rule A: If client's risk profile is risk adverse
Then client's investment strategy is blue chip stocks

Rule B: If client's investment portfolio is less than $50,000
And client's age is more than 60
Then client's investment strategy is high-grade bonds

Rule C: If client's risk profile is risk taker
And client's age is less than 45
Then client's investment strategy is growth stocks

In essence, we have concluded that either strategy is advisable. Thus in this case, of deterministic rule bases, the validity of multiple conclusions is a function of the situation. Again, however, realize that not all development packages permit multiple conclusions.

CHAPTER 4

KNOWLEDGE ACQUISITION

Definition of knowledge acquisition as transfer and transformation of potential problem solving the expertise from some knowledge source to a program.

Knowledge acquisition is a generic term, asit is neutral with respect to how the transfer of knowledge is achieved. The knowledge elicatation, on the hand, often implies that transfer isaccomplished by a series of interviews between a domain expert and a knowledge engineer who then writes a computer program representing the knowledge.

The term could also be applied to the interaction between an expert and a program whose purpose is

- To elicit knowledge from experts in some systematic way.
- To store knowledge so obtained in some intermediate representations.
- To compile the knowledge from the intermediate representation into a runnable form, such as production rules.

The use of such programs is adventageous because it is less labor intensive, and because it accomplishes the transfer of knowledge from the expert to a prototype in a single step.

4.1 THEORITICAL ANALYSIS OF KNOWLEDGE ACQUISITION

Knowledge eliction interviewsgenerate between two and five production rule equivalents per day. The reasons why productivity is so poor include

- The technical nature of specialist fields requires the non–specialist knowledge engineer to learn something about the domain before communication can be productive,
- The fact that experts tend to think less in terms of general principles and more in typical objects and commonly occuring the events,

- The seach for good notation for expressing the domain knowledge, and a good framework for fitting it all together, is itself hard problem, even before one gets down to the business of representing the knowledge in a computer.

4.2 STAGES OF KNOWLEDGE ACQUISITION

It is worth summarizing these stages are here:

- **Identification:** Idetify the class of problems that the system will be expected to solve, including the data that the system will work with, and the criteria that solutions must meet. Identify the resources available for the project, in terms of expertise, manpower, time constrains, computing facilities and money.
- **Conceptualization:** Uncover the key concepts and the relationship between tehem. This should include a characterization of the differnt kinds of data, the flow of information and the underlying structure of the domain, in terms of causal spatio-temporal, or part-whole relationships, and so on.
- **Formalization:** Try the understand the nature fo the underlying search space, and the character of the search that will have to be conducted. Important issues include the certainty and completeness of the information, and other constraints upon th elogical interpretation of the data, such as time dependency, and the reliability and consistency of the differnt data sources.
- **Implementation:** In turning a formalization of knowledge into runable program , one is primarily concerned with the specificataion of the control and the details of information flow.
- **Testing:** The evalution of expert systems is far from being a exact science, but it is clear that the taskcan made easierif one is able to run the program on a large and representative sample of test cases. Common sources of error are rules which are either missing , incomplete, or wholly incorrect, while competition between related rules can be cause unexpected bugs.

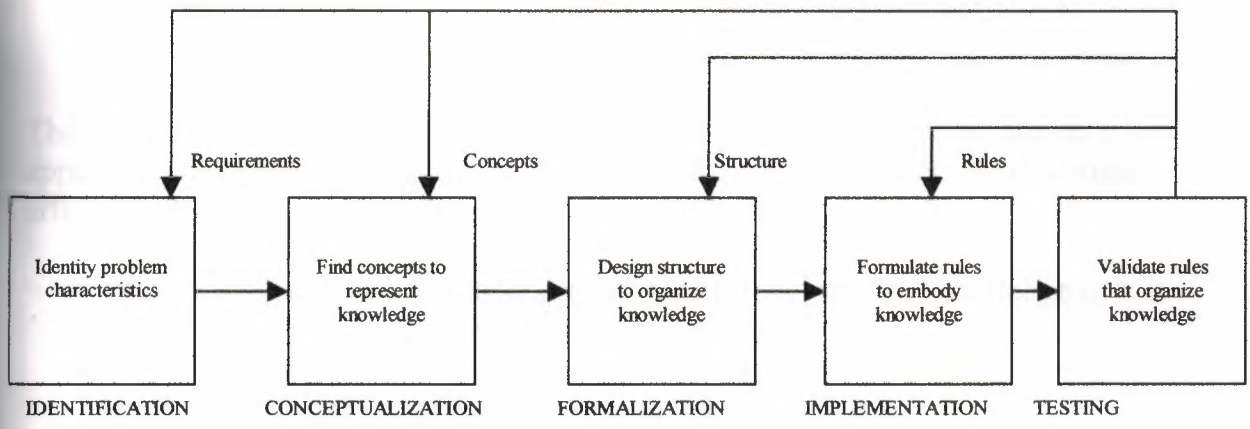


FIGURE – Stages of knowledge acquisition

4.3 DIFFERENT LEVELS IN THE ANALYSIS OF KNOWLEDGE

The distinction drawn between identification, conceptualization and formalization can also be found who have developed a modeling approach to knowledge engineering within frame called KADS. The autors argue that a knowledge-based system is not a container filled with knowledge extracted from an expert but an " operational model " that exhibits some desired behaviour and impacts real world phonemena. Knowledge acquisition involves not just eliciting domain knowledge but also interpreting the elicited data with respect to some conceptual framework and formalizing these conceptualizations in such a way that a program can actually use the knowledge.

The KADS framework is founded on five basic principles, as follows:

1. The introductio of multiple models as a means to cope with the complexity of the knowlege engineering process.
2. The KADS four-layer framework for modeling the required expertise.
3. The reusability of generic model components as templates supporting top-down knowledge acquisition.
4. The process of differentiating simple models into more complex ones.
5. The importance of structure-preserving transformation of modelsof expertise
   into design and implementation.

Today's knowledge engineer is faced with a large space of methods, techniques and tools which could be used to build an expert system. However, he or she is also faced with three in variant issues, namely

- Defining the problem that the expert system is meant to solve,

- Defining the function that the expert system will fullfill with respect to that problem,
- Defining the tasks that must be performed in order to fullfill that functions,

They made a slightly different set of distinctions level of analysis, which now appear to overlap with the models proposed in KADS-II, but are nonetheless worth articualiting,

- Knowledge conceptualization aim atthe foraml descripion knowledge in terms primitive concepts and conceptual relations.
- Epistemological analysis is concerned to uncover the structural properties of the conceptual knowledge, such as taxonomic relations.
- Logical analysisis concerned with the knowledge about how to perform reasoning tasks in the domain.
- Implementational analysis deals with the mechanisms upon which other levels of analysis are based.

4.4 ONTOLOGICAL ANALYSIS

Another knowledge – level analysis for expert problem solving is called ontological analysis. This approch describes systems in terms of entities, relations between them, and transformations between entities that occur during the performance of some task. The authors use three main catagoriesfor structuring domain knowledge:

- The static ontology, which consists of domain entities, together with their propertities and relations,
- The dymamic ontology, which defines the states that occur in problem solving, and themanner in which one state may be transformed into another,
- The epistemic ontology, which describes the knowledge that guides and constrains state transformations.

There is less of correspondence with lower levels, such as the logical and implementational analyses.

4.5 EXPERT SYSTEM SHELL

Early expert systems were built "from scratch", in the sense that the architects either used the primitive data and control structures of an existing programming language to represent knowledge and control its application, or implemented a special purpose rule or frame language in an existing programming language, as a prelude to representing knowledge in that special purpose language.

- Modules, such as rules or frames, for represeting knowledge,
- An interpreter which controlled when such modules became active.

The modules, taken together, constituted the knowledge base of the expert system, while the interpreter constituted the inference engine. In some cases, it was clear that these components were reusable, in the sense that they would serve as a basis for other applications of expert system technology. Since such programs were often abstractions of existing expert systems, they became known as expert system shells.

4.6 KNOWLEDGE ACQUISITION METHODS

One involves knowledge acquisition for troubleshooting a telephone company switching system, and the other involves planning therapeutic regimes for cancer patients. The two projects dealt with the issues of knowledge acquisition and knowledge representation in rather differnt ways, largely as a consequence of both the task at hand and the way that the task was appraached.

4.6.1 KNOWLEDGE ELICTION BY INTERVIEW IN COMPASS

A telephone company " switch" is not simple device, but extremely complex system whose circuitry may occupy a large part of building. The goals of switch maintenance ar eminimize to the number of calls that have to be rerouted owing to bad connections and ensure that faults are repaired quickly to maintain the redundancy of the system. Bad connections are caused by some failure in the electrical path through the switch that connects two telephone lines.

COMPASS is an expert system which examines error messages derived from the switch's self test routines, which look for open circuits, short, lag time, in the operation of components, and so forth. The cause of a switch problem can only be identified by looking at a series of such messagesand bringing significant expertise to bear. COMPASS can suggest the running of additional tests, of the replacement of a particular component, such as a relay or circuit card.

The knowledge acquisition cycle employed in COMPASS had the following form:

1. Elicit knowledge from the expert.
2. Document elicited knowledge.
3. Test the new knowledge as follows:
    - Have the expert anlyze a new set of data.

- Analyze the same data in a hand simulation using the documented knowledge.
- Compare the results of the expert's opinion with the hand simulation.
- If the results differ, then find the rules or procedures that generated the discrepancy, and the return to (1) to elicit more knowledge from the expert to resolve the problem, else exit loop.

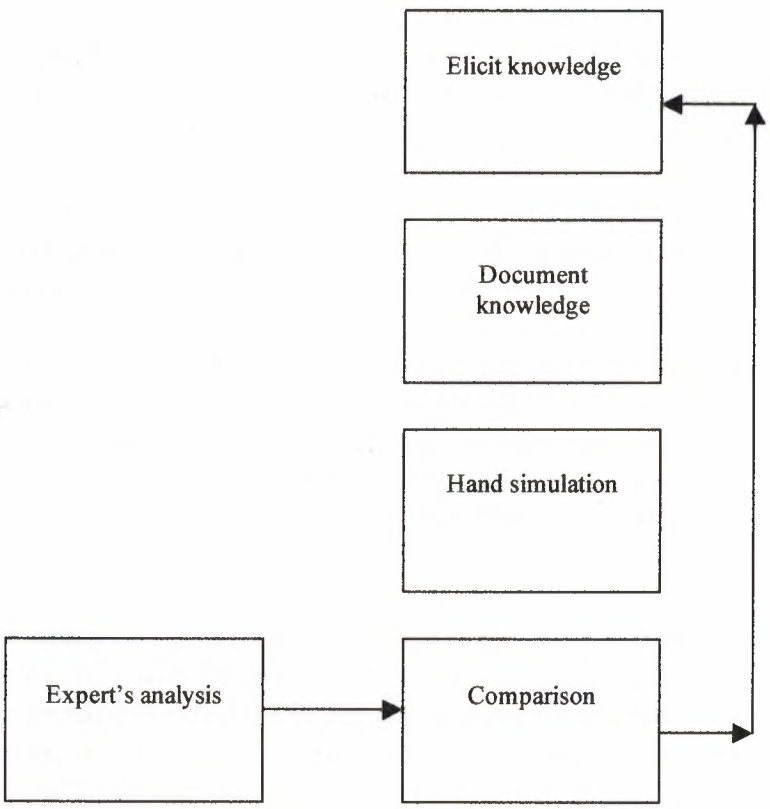


FIGURE – Knowledge acquisition cycle in COMPASS.

This elicit-document-test cycle is represented graphically in above figure.

4.7 KNOWLEDGE – BASED KNOWLEDGE ACQUISITION

We shall see that many of the learned from trying to extend expert system technology in various directions have application to knowledge acquisition problem. Specifically,

- Attempts to use expert systems as a basis for intelligent tutoring ssytems have led to a deeper understanding of the differnt kinds of knowledge that experts deploy in problem solving; and
- Attempts to build generic expert system tools like EMYCIN have posed interesting problems concerning how to help developers with the task of encoding knowledge from some arbitrary domai into frame or production rule format.

Such endeavors have required researchers to examine the role of domain knowledge and domain inference more closely, particularly with respect to the diffrent styles of reasoning that are approximate to diffrent domains.

Looking ahead slightly, what seems to be clear is that knowledge acquisition is greatly facilitated by beingitself knowledge based. In other words, a knowledge elitation programs needs some knowledge of a domain or a problem area ın order to acquire new knowledge effectivel, just as knowledge engineers need to have some knowledge of a domain before they can communicate effectively with an expert.

Perhaps this result is not suprising, given the lessons of knowledge-based approaches to problem solving. Knowledge elicitation is a substantial problem in itself, and there is no reason to suppose that there is a single general method that will be effective in all domains, any more than there is reason to suppose that there are general problem solving methods that will always be effective.

Knowledge elicatation model by interview based on a domain model is not the last word in automated appraaches to acquisition. We shall two further approaches:

- Acquisition strategies organized around a particular problem solving method,
- Unsupervised machine learning of rules by induction.

4.8 KNOWLEDGE ACQUISITION AND THE DOMAIN EXPERT

It woulld seem taht the most obivious in which one may acquire a knowledge base is to go directly to the human expert. However, there are at least four rasons why this may not work, or at least not provide tatally satisfactory results.

- For some problems, there simply may not be an expert. One example that comes to mind is that of investing in the stock market. While some inventor or investment advisory services do well for a relatively brief time., they typically have spells in which their performanceis mediocreto terrible.
- To alleged experts may actually be exhibiting poor mediocre performance. All too often, the term expert is loosely applied to anyone who simply gets the job done.
- The expert may not wish to reveal their tricks of the trade. In some cases, such individuals simply refuse to cooperate. In others, potentially far more serious problems occurs.
- The experts may not wish who are just unable articulate the approach that they use. Many experts, in fact, simply and honestly do not really understand how they actually make their decisions.

However, bssed upon the assumption that a human is performing the task that is to be performed in the future by the expert system, our first step is to identify that person. Once this person has been identified, the next step is to set up to an initial meeting with the alleged domain expert. This meting should be informal beause yuo must decidedly want the inaugural meeting with this person to take place in a relaxed atmosphere.

There are several purposes for the initial meeting. First, we wish to relax the individual. Second, we should attempt to explain to the individual just percisely what it is that we intend to accomplish. Typically, one emphasizes that our purpose is not to use and then discard the expert, but rather it is to provide him or her with a computerized assistant so that he or she can pursue more intering work.

In certain cases, the initial meeting should be followed, or even preceded by an on site visit. There is simply no substitute for actully being able to view the problem in its physical context.

There is yet another purpose to the initial meeeting, as well as those that follow, that should be openly discussed. Specifically, we should use this meeting, as well as follow-on metings, to attempt to evaluate the true extent of the expertise of our expert.

If and when you encounter a domain expert in whom youm have no confidence, there are number of alternatives that should be considered. First, and most obviously, you might try to find a replacement - someone else in the organization who seems reosanably competent in the domain under consideration. This option of course, reqires a certain amount of diplomacy. Second, you might consider learning enough about the problem at hand so that you can act as the domain expert. Third, might wish to examine historical records of decision made.

Returning to primary point our discussion, as long as feel confident that the expert systems approach is indeed the most appropriate, and the domain expert is reasonably competent, wemay continue with the knowledge acqusition process. Thus, following initial informal meeting with the domain expert, we shold conduct a series of formal meetings designed to extract as much information.

The conduct of the follow-on meeitings is generally best handled through the employment of two knowlege engineers. And at least oene of these individuals should be experienced. One knowledge engineer should be given the primary responsibility for conducting the interview, while the other knowledge engineer listens carefully to both the questions and responses. The second knowledge engineer will also make sure that the meeting is being properly recorde and, when necessary, replace tapes and move microphones.

At least one of the knowledge engineers shold be experienced in knowledge acquisition and the successful of development of expert systems. One of the worst mistakes being made in expert system developmentis the assignment of inexperienced personnel to the effort.

One must always kep in mind that the purpose of these meetings is to extract the knowledge base of the expert. While this sounds obvious, it has been observed that the discussions in some knowledge acquisition meetings meander off onto tangents as the discussants pursoe points that have little if any bearning on the knowledge base.

There are several modes through which the knowledge basemay be extarcted during the such meeting. One is to simply ask the expert system to expalin the procedure through which he or she arrives at a conclusion. Another approach is to conduct demonstration sessions wherein the expert is asked to proceed through the decision making processfor a series of examples. In general, the second approach lends itself better to knowledge acquisition.

One of the practices that we employed, with considerable success, is to ask the domain expert to go through a demonstration of the decision making process. – at

our office. We have found that is an excellent way to determine just what data the domain expert actually requires decision making.

At the conclusion of each session, the knowledge engineers will typically try to restate the responses of the expert in the production rule format. Thus, after a few sessions, it should be possible to develop a simple, prototype expert system. Once prototype is available, we may use it to extract additional knowledge from the expert.

There are a number of good papers that discuss the conduct of the knowledge acquisition phase. In particular, we have provided some excellent guidelines for knowledge acquisition. Here, we have attempted to summarize our throughts on knowledge acquisition, where the influenceof numerous other authors is acknowledged. These guidelines are presented in the list that follows.

4.8.1 SELECTION OF THE DOMAIN

- The domain should be one for which the expert systems approcah is truly approppimate, and for which expert system would provide some dstinct advantage over any alternative methods.
- Good decision making within the domain should be of succifient importance to management that they are willing to commit the timeand resources necessary to support the development and implementation of expert system.
- Management must recognize both the costs and risks of an expert systems development knowledge engineers, over a reasonable period of time.
- The domain should be relatively stable; in particular, dramatic changesover the period of the development effort should not be foreseen.

4.8.2 SELECTION OF THE KNOWLEDGE ENGINEERS

- Ideally, two knowledge engineers should be used, where at least one of these is experienced in development and implementation (successful) expert systems.
- The knowledge engineers should not be one thick ponies. That is, they should at the very least be aware of alternative approaches to decision analysis.
- The primary skills of the knowledge engineers should be in the areas of eleiciting knowledge and forming a model oof that knowledge.

4.8.3 SELECTION OF THE EXPERT

- Ask the organization to provide you with the names of candidate domain experts, that is, those individuals who are believed to have significant expertise within the domain in question.
- Select a domain expert whose performance is generally acknowledgedto be above and beyond that the most others performing the task.
- Select an expert with successful track record over a period of time.
- Select an expert who is both willing and able to comunicate personal knowledge, and who relatively articulatein doing so,
- Select an expert who is both willing and abl to devote thetime necessary to support the development effort.
- If no expert can be idendified, or made available, consider the development of the rule base through alternative means.

4.8.4 THE INITIAL MEETING

- Prior this meeting , theknowledge engineers should make an all-out effort to familiarize themselves with the problem, the domain, and the terminology used within the domain.
- Locate this meeting in comfortable surroundings.
- This meeting should be conducted in a informal, relaxed manner.
- Tell the expert what your plans and goals are, and explain just what an expert system is and what it can do (cannot do) for the expert as well as the organization.
- Explain the evaluatin of the expert system.
- Reinforce your discussion of expert systems with the demonstration of the use of some existing expert system. However, avoid the demonstrations of an expert system that is all too obviously a toy.
- If audio/visual recording is desired, ask the expert for permission to do so – and explain that these recordings will befor the private use of the knowledge engineering team.

4.8.5 ORGANIZATION OF FOLLOW – ON MEETINGS

- Attempt to minimize the possibility of interruptions. Set aside meeting times during the which the expert can devote his or her full attention to the effort.
- Establish a formal agenda for each meeting.
- Establish goals and objectives fr each meeting.

- Onceprototype expert system has been developed, establish access to the supporting software and hardware.

4.8.6 CONDUCT OF THE FOLLOW – ON MEETINGS

- Elicit the rules through discussion and demonstration.
- Attempt to identify all external sources of data and information that are used by the expert.
- Be patient. Don't interrupt the domain expert.
- Avoid criticism – instead, focus on clarification.
- Always rememberthat you are building a model of the expert's rule base, not a model of your rule base.
- If you don't understand a point made by the expert, don't be afraid to admit it. Ask for clarification.
- Use test cases to both demonstrate the decision making process and identify the limits over which the rule based is valid.
- Acquaint the domain expert with production rules; this may encourage the expert to being stating his or her rules in this format.
- Always remember what you are there for.

4.8.7 DOCUMENTATION

- Document the results of the meeting as soon as possible after the meeting (preferably, immediately after the meeting)
- Documentation for each meeting should include such facts as:

    - Date, time, and location for meeting.
    - Name of expert.
    - List and description of the rules identified during the meeting.
    - List of any new objects, attributes and/or values encountered – their properties.
    - Identification of any new outside sources and references.
    - Listing of new terminology encountered, and associated definitions.
    - Listing and discussion of any gaps or discrepancies encountered.
    - Remainders.

- Documentation in support of all production rules thus far developed should include such facts as:

    - A listing and description of all rules thus far developed.

- A listing and description of all objects, attributes, and values thus far encountered.
- Source and reference list
- Glossary of domain terminolgy
- Listing and discussion of the test cases used to evaluate the prototype.

4.8.8 MULTIPLE DOMAIN EXPERT SYSTEMS

One additional consideration in knowledge based development: the existence of more than one domain expert. Some authors have noted that this situation can be particularly frustrating if not properly and delicately handled. However, he advisesthat the knowledge engineer need not be particularly concerned about multiple experts. That is, using a rule base cloned from one expert, we build a prototype expert system and then let the other domain experts critique results.

Our experiences in dealing with multiple experts has followed similar approach. We have always selected on domain expert as the individual from whom the rules were to be acquired, that is, as the key expert.We have presented the protypes to the remaining experts for a critique. In doing this, we have tried to discourgage the key expert from attending such presentations. We feel that his or her attendance may cause the other experts to feel less free making their commentsand criticisms.

There is yet another situation in which multiple experts may be encountered. However, rather than having mastery across the entire domain of interest, these experts may each have expertise in various portions of the doman. One approach to this situation is to develop a set of expert systems, one for each subdomain. Another is to utilize separate knowledge basis and to coordinate these through asingle expert systems package – by means of the blackboarding approach. And this precisely the approach used in HEARSAY.

CHAPTER 5

EXPERT SYSTEMS FOR MEDICAL DIAGNOSIS

In this chapter, we consider the implementation of expert system for medical diagnosis of human illness. As, an example diagnosis of stomach disease is considered. To develop expert system for this problem we collect knowledge from experienced specialists and different references. On the base of collected information, we have created knowledge base for diagnosis stomach diseases. Knowledge base consists of two parts: diagnosis and recommendation. In diagnosis part, knowledge base has production rules that have premise and conclusion parts. Premise part of expert system includes system inputs. Those inputs are: level of indigestion after eating meal, lack of appetite, pain, laboratory investigations etc. The conclusion part of diagnosis includes the name of diseases (gastritis, stomach ulcer and stomach cancer and their different levels). After defining disease, expert system makes recommendation for its treatment. In this case, knowledge base of premise part includes levels of diseases defined in the result of diagnosis (gastritis, stomach ulcer and stomach cancer). Conclusion part gives recommendation for treatment of diseases. In table-1 and table-2, the knowledge base for diagnosis and recommendation for treatment of stomach diseases are given.

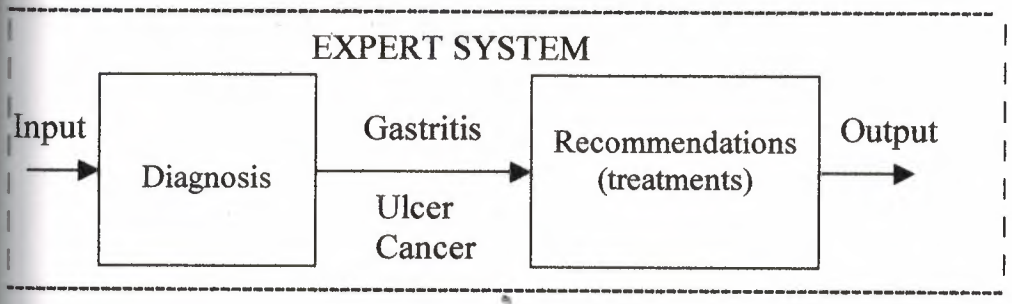


FIGURE-1. Simple diagram of an expert system for medical diagnosis.

This expert system will behave as a doctor. And it checks inputs according to diagnosis (i.e. according to knowledge) and using this knowledge base, it will determine disease and recommendations for its treatment.

DIAGNOSIS		
GASTRITIS	STOMACH ULCER	STOMACH CANCER
1. What are the symptoms? • Indigestion after meal. • He/she can have a sensitivity zone on the belly above the navel. • Lack of appetite, fullness upper side of the belly zone, he/she can have a complaint such as ulcer. • Sometimes, nausea and vomiting. 2. In the laboratory investigations, which result are observed? • Laboratory investigations are usually normal. • For atrophic gastritis, they can determine the inadequacy of acid in the water stomach. • For chronic hypertrophic gastritis, there is an action in the stomach. 3. What are the negative effects of soup water and acid in the stomach and the intestine that children drink by mistake? • It can pierce in the stomach.	1. What are the symptoms? • After meal and before taking the meal, pain belly above navel. • Sensitivity on the belly, above navel can be found with the hand pressure. • Result of bleeding of the ulcer can be observed anemia. • Sometimes, pain can spread to me left side of belly. • At more high intensity, with vomiting, relief after vomiting and loss of weights.	1. What are the symptoms? • In the belly, weight and swelling. • There are symptoms such as complaints of stomach ulcer. • After swallowing hard foods, there is a discomfort and feeling of swelling. • A mass can be found by feeling the belly with hand. • Colour of faeces is deep black accompanied by anaemia. • Results of vomiting, contents of this come form of coffee grounds.

GASTRITES	STOMACH ULCER	STOMACH CANCER
• Difficulty in swallow can be observed. • Intense burning on the zone of the belly. • There is pain like cramp. • There is nausea, diarrhoea and vomiting. Vomiting is usually bloody.		

TABLE-1: This table shows main diseases of the stomach and their diagnosis.

RECOMMENDATIONS (TREATMENTS)		
GASTRITIS	STOMACH ULCER	STOMACH CANCER
1. What are the treatments of gastritis? • There is a diet prescribed by doctor. • Refrain matters that increased complaints. 2. What are the treatments of the negative effects of soup water and acid in the stomach and the intestine that children drink by mistake? • He/she must take drug by doctor that causes vomiting. • Stomach wash. • Intravenous liquids • He/she can be given antacid treatment with the way of mouth.	1. How can the stomach ulcer be treated? • Beginning medical treatment. • If medical treatment does not give result, he/she will be treated by surgical operation. 2. Foods that should be avoided ulcer? • Meat soup/gravy, water of meat, peppery, vinegary and spices foods, pickle, orange, lemon, melon, tinned food, dried fruits, and fried foods in oil. • Tea, coffee, drinks (that have carbonate) • Alcoholic drinks • Cigarette 3. What are the drugs that should not be used by those having ulcer? • Aspirin • Drugs that decrease pain of rheumatism. • Corticosteroids and ACTH Phenacetin	1. What are the treatments of stomach cancer? • Surgical operation.

TABLE– 2: This table shows recommendations (treatments) of main disease of the stomach.

Here, there are three type of diet of stomach ulcer for treatments.

DIET OF ULCER (I)	
GROUP OF FOODS	FREE FOODS
Drinks	Milk, linden, garden sage, drink made with sweet and fresh yogurt.
Meats, fishes and poultry	All of these are prohibition.
Grain and their products	White bread, biscuit, cracker, rice, macaroni, vermicelli, hardtack.
Egg and cheese Soups	Hard egg, cheese (no salt), flour, flour of pea and lentil, soup made of puree of vegetable.
Vegetables and their waters	Puree of vegetable (pea, carrot, squash, green beans, spinach, potato)
Fruits and their waters	Ripe banana, cooked with peeled apple and peach as form of stewed fruit.
Oils	Butter, olive oil, sunflower oil, margarine, corn oil.
Desserts	Sugar, honey, jam (without grain), pudding, rice pudding, cream, simple Turkish delight.
Foods (taste)	Light salt, sauces using milk and flour.

DIET OF ULCER (II)	
GROUP OF FOODS	FREE FOODS
Drinks	Milk, linden, garden sage, drink made with sweet and fresh yogurt.
Meats, fishes and poultry	Meat of calf, sheep, lamb, chicken, cattle, turkey and fish (all of these are billed or grilled).
Grain and their products	White bread, cracker, simple biscuit, simple sugared dried pastry, pie of thin layer of dough, macaroni, vermicelli, soup of semolina.
Egg and cheese	Soft boiled or hard egg, white cheese, sheep cheese.
Soups	Filtered vegetable soups, flour soup, vermicelli, rice, and plateau soups (without water of meat).
Vegetables and their waters	Well-cooked carrot, climbing kidney beans, squash, beet, purslane, chard, potato, water of carrot and tomato.

Fruits and their waters	Ripe banana, cooked with peeled apple and peach as form of stewed fruit.
Oils	Natural oils (butter, olive oil, sunflower, margarine, corn oil that have small amount acid).
Desserts	Sugar, filtered honey, jam, pudding, rice pudding, simple Turkish delight, jelly, desserts with jelly, pudding made of rice flour and shredded chicken.
Foods (taste)	Flour, milk, salt, and sauces made of oil.

DIET OF ULCER (III)	
GROUP OF FOODS	FREE FOODS
Drinks	Milk, milk with banana, linden, garden sage, yogurt made in home, drink made of yogurt, tea, lemonade, milky coffee.
Meats, fishes and poultry	Meal of calf, cattle, sheep, chicken, turkey, liver, kidney, spleen that are boiled or grilled.
Egg, cheese, grain and their products	All of these are free.
Soups	All kind of soups made of cooked in simple water and water of tomato.
Vegetables	All of cooked vegetables, water of tomato and carrot.
Fruits and their waters Desserts	All of fruits and their waters are free, simple cake, pudding, rice pudding, cream, honey, jam, jelly, stewed fruits, deserts with jelly, simple Turkish delights, grape molasses, dessert of pumpkin.
Oils	All of these are free.
Foods (taste)	Salt, creams, allspice, cinnamon, dessert red paper, thyme, mint, cumin, olive.

	GASTRITIS		STOMACH ULCER	STOMACH CANCER
	NON-CHRONIC	CHRONIC		
PAIN	After meal Small	After meal Small (Indigestion) very often	Before and after meal Large	Very Large
SITIVITY	Small	Middle	Large	Very Large
USEA & MITING	Small (Sometimes nausea)	Small	Large (At more intensity)	Very Large
EEDING	No	Small	Middle (Anemia)	Large

LE-3: The general diagram for diagnosis.

Using table-1, we can generalize like table-3 for knowledge base. The most ortant advantage of preparing this table (like table-3) is provided to write luction rules easily for expert system.

There are 256 combinations. That means, there are 256 production rules for rt system. Some of these rules are below:

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = SMALL OR (NAUSEA OR VOMITING)= SOMETIMES
AND BLEEDING= NO
THEN DISPLAY (" NON- CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = SMALL OR VOMITING= SOMETIMES
AND BLEEDING=SMALL
THEN DISPLAY (" NON-CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = SMALL OR (NAUSEA OR VOMITING)= SOMETIMES
AND BLEEDING= MIDDLE OR BLEEDING= ANEMIA
THEN DISPLAY (" NON-CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = SMALL OR (NAUSEA OR VOMITING)= SOMETIMES
AND BLEEDING=LARGE
THEN DISPLAY (" NON-CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = SMALL
AND BLEEDING=NO
THEN DISPLAY (" NON-CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = SMALL
AND BLEEDING=SMALL
THEN DISPLAY (" CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = SMALL
AND BLEEDING= MIDDLE OR BLEEDING= ANEMIA
THEN DISPLAY (" CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = SMALL
AND BLEEDING=LARGE
THEN DISPLAY (" CHRONIC GASTRITIS ");

```
IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = LARGE OR (NAUSEA OR VOMITING)=AT
MORE HIGH INTENSITY
AND BLEEDING=NO
THEN DISPLAY (" NON-CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = LARGE OR (NAUSEA OR VOMITING)=AT
MORE HIGH INTENSITY
AND BLEEDING=SMALL
THEN DISPLAY (" CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = LARGE OR (NAUSEA OR VOMITING)=AT
MORE HIGH INTENSITY
AND BLEEDING= MIDDLE OR BLEEDING=ANEMIA
THEN DISPLAY ("ULCER");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = LARGE OR (NAUSEA OR VOMITING)=AT
MORE HIGH INTENSITY
AND BLEEDING=NO
THEN DISPLAY (" NON-CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = LARGE OR (NAUSEA OR VOMITING)=AT
MORE HIGH INTENSITY
AND BLEEDING=LARGE
THEN DISPLAY ("CANCER");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = VERY LARGE
AND BLEEDING=NO
THEN DISPLAY (" NON-CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = VERY LARGE
AND BLEEDING=SMALL
THEN DISPLAY (" CHRONIC GASTRITIS ");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = VERY LARGE
```

```
AND BLEEDING= MIDDLE OR BLEEDING= ANEMIA
THEN DISPLAY ("ULCER");

IF PAIN= SMALL
AND SENSITIVITY= SMALL
AND (NAUSEA OR VOMITING) = VERY LARGE
AND BLEEDING=LARGE
THEN DISPLAY ("CANCER");

IF PAIN= SMALL
AND SENSITIVITY= MIDDLE
AND (NAUSEA OR VOMITING) = SMALL OR (NAUSEA OR VOMITING)=
SOMETIMES
AND BLEEDING=NO
THEN DISPLAY ("NON CHRONIC GASTRITIS");

IF PAIN= SMALL
AND SENSITIVITY= MIDDLE
AND (NAUSEA OR VOMITING) = SMALL OR (NAUSEA OR VOMITING)=
SOMETIMES
AND BLEEDING=SMALL
THEN DISPLAY ("CHRONIC GASTRITIS");

IF PAIN= SMALL
AND SENSITIVITY= MIDDLE
AND (NAUSEA OR VOMITING) = SMALL OR (NAUSEA OR VOMITING)=
SOMETIMES
AND BLEEDING=MIDDLE OR BLEEDING= ANEMIA
THEN DISPLAY ("CHRONIC GASTRITIS");

IF PAIN= SMALL
AND SENSITIVITY= MIDDLE
AND (NAUSEA OR VOMITING) = SMALL OR (NAUSEA OR VOMITING)=
SOMETIMES
AND BLEEDING=LARGE
THEN DISPLAY ("CHRONIC GASTRITIS");

IF PAIN= SMALL
AND SENSITIVITY= MIDDLE
AND (NAUSEA OR VOMITING) = SMALL
AND BLEEDING=NO
THEN DISPLAY ("CHRONIC GASTRITIS");

IF PAIN= SMALL
AND SENSITIVITY= MIDDLE
AND (NAUSEA OR VOMITING) = SMALL
AND BLEEDING=SMALL
THEN DISPLAY ("CHRONIC GASTRITIS");
```

CONCLUSION

The graduation project is devoted to developing of expert system for medical diagnostics. To solve given problem the application of different expert systems for solving diagnostics problems are given. The expert system concepts, its characteristics, comparison of expert system and DSS, description of different expert system are considered. The effectiveness of expert system for medical diagnosis problem solving was shown.

After the structure of expert system and the functions of its main blocks are described. The representation models of knowledge, such as OAV triplets, semantic networks, frames, neural networks, predicate logic, rule-based systems are described. As an example for given problem rule-base model is chosen. The main components of rule-base model, its structure, attribute-value property, clause property, rule property are widely described.

One of the main blocks of expert system knowledge acquisition is analysed. Stages of knowledge acquisition, different level of analysis of knowledge, different way of knowledge acquisition are explained.

The diagnostics of stomach and intestine diseases is considered. Using experienced experts' knowledge and different medical references the knowledge base that has production rules, for diagnostics of the stomach diseases is created.

Knowledge base has about 256 production rules. Premise parts of rules include the input (characteristics) features of illness, the conclusion part – the defined illness. After defining illness the recommendation for its treatment is given. The created knowledge base is used in ESPLAN expert system shell (developed in Azerbaijan State Oil Academy) to solve given problem. The received results satisfy the efficiency of the applied approach.

REFERENCES

1. AI WEEK, " Big Eight Accounting Firm Dvevelops Tax ES," AI WEEK, July 1, 1988a,p.4.
2. Rafik Aliyev, Fuad Aliyev and Mamedgasan Babaev. Fuzzy Process Control and Knowledge Engineering in Petrochemical and roboticManufacturing, Verilag TUV Rheinland, 1991, p.146.
3. Aikens, J. S., J. C. Kung, E. H. Shortliffe, and R. J. Fallat, " PUFF: An Expert System for Interpretation of Pulmonary Function Data," in B. C. Clancey and E. H. Shortliffe, (eds.), redings in Medical Artificial Intelligence: The First Decade, Addison-Wesley , Readings, Mass., 1984
4. Anderson, D., and C. Ortiz, "AALPS: A Knowledge-Based System for Aircraft Loading," IEEE Expert, Winter 1987, pp. 71-79.
5. Benchimol, G., P. Levine, and J. C. Pomerol, Developing Expert Systems for Business, North Oxford Academic, London, 1987.
6. Bonissone, P. P., and H. E. Johnson, " Expert System for Diesel Electric Locomotive Repair,"
7. Brazile, R. P., and K. M. Swigger, " GATES: An Airline Gate Assignment and Tracking Expert System, " IEEE Expert, Summer 1988, pp. 33-39.
8. Carter, C., and J. Catlett, " Assessing Credit Card Applications Using Machine Learning," IEEE Expert, vol. 2, no. 3, Fall 1987, pp. 71-79.
9. Casey, J., "Picking the Right Expert System Applications," AI Expert, September 1989, pp. 44-47.
10. Cavalier, T.M., J. P. Ignizio, and A. L. Soyster, "Discriminant Analysis via Mathemetical Programming: On Certain Problems and Their Causes," Computers and Operations Research, vol. 16, no. 4., 1989, pp. 71-79.
11. Feigenbaum, E. A., Interview in Knowledge-Based Systems: A Step-by-Step Guide to Getting Started, Second Annual AI Satellite Symposium, Texas Instruments, 1986.
12. Huntington, D., EXSYS Expert Systems Development Package, EXSYS Manual, Albuquerque, New Mexico, 1985.
13. Ignizio, J.P., " Identification of Incompleteness in Knowledge Bases via a Rule Dependency Maxtix," technical paper, Department of Industrial Engineering, University of Houston, july 1987b.
14. Ignizio, J.P.," Attribute-Value Pair Tables and Rule Base Architecture," technical paper, University of Houston, Houston, Tex., 1988.
15. Lindsay, R.K., B.G., Buchanan, E.A. Feingenbuam, and J. Lederberg, Applications of Artificial Intelligence for Chemical Inference: The DENDRAL Project, McGraw-Hill, New York, 1980.
16. Pople, H. E., jr., "CANDUCEUS: An Experimental Expert System For Medical Diagnosis," in P. H. Winston and K. A. Prendegast (eds.), The AI Business, M.I.T., Cambridge, Mass., 1984 pp. 67-80.
17. Reddy, D. R., L. D. Erman, R. D. Fennell, and R. B. Neely, " The HEARSAY Speech Understanding System: An Example of the Recognition Process," IJCAI-3, 1973, pp. 185-193.
18. Shortlife, E. H., Computer-Based Medical Consultations: MYCIN, Elsevier, New York, 1976.
19. James P. Ignizio, Introduction To Expert System – The Development And Implementation Of Rule-Based Expert Expert System, McGraw Hill International Editions, Computer Science Series 1991.

20. Peter Jackson, Introduction To Expert System, Addison Wesley Longman Limited 1999.
21. Prof. Dr. Engun Göney, " 1000 Question 1000 Answer Health Problems – Diseases of Stomach and Intestine (in Turkish language, 1000 Soru 1000 Cevap Sağlık Sorunlarınız – Mide-Bağırsak Hastalıkları) ," Milliyet, june 1990.
22. R. A. Aliev, N. B. Abdikeiev, M. M. Shakhnazarov. Manufacturing Systems with Artifcial Intelligence, Radio 1 Sviaz, Moscow, 1990.
23. R. A. Aliev, R. H. Abiyev, R. R. Aliev. Industrial Controllers Based On Neural Expert System. //World Congress on Expert Systems, Estoril/Lissabon, Portugal, 1994.