UNIT-V EXPERT SYSTEM

Expert systems-Architecture of expert systems, Roles of expert systems-Knowledge Acquisition– Meta knowledge, Heuristics. Typical expert systems-MYCIN, DART, XOON, Expert systems shells.

1. EXPERT SYSTEM-BASIC CONCEPTS

What is Expert System?

- An expert is a person who has the expertise and knowledge of his specialized field. Heart specialist and a mathematics expert etc
- Through experience, an expert expands his skills to enable him to solve problems heuristically, efficiently and effectively.

What is Expertise?

- Expertise is the extensive, task-specific knowledge acquired from training, reading and experience
- Enables experts to be better and faster than non experts

Definition 1:

Expert System (ES) is a computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require human expertise for their solutions.

Definition 2:

Expert System (ES) is an information system that is capable of mimicking human thinking and making considerations during the process of decision-making

History of Expert System

Early to Mid-1960s

- One attempt: the General-purpose Problem Solver (GPS)
- General-purpose Problem Solver (GPS)
- A procedure developed by Newell and Simon from their Logic Theory Machine -
    - Attempted to create an "intelligent" computer
    - Predecessor to ES
    - Not successful, but a good start
    - First Expert system was DENDRAL by stanford university in the late 1960s

Mid 1970s

- Several Real Expert Systems Emerge
- Recognition of the Central Role of Knowledge
- AI Scientists Develop
- Comprehensive knowledge representation theories
- General-purpose, decision-making procedures and inferences
- Limited Success Because
- Knowledge is Too Broad and Diverse
- Efforts to Solve Fairly General Knowledge-Based Problems were Premature

Early 1980s

- ES Technology Starts to go Commercial
- XCON, XSEL
- CATS-1
- Programming Tools and Shells Appear
- EMYCIN, EXPERT

- META-DENDRAL
- EURISKO
- About 1/3 of These Systems Are Very Successful and are Still in Use

Need of Expert System

- An Expert System is built because of two factors: either to replace or to help an expert.
- Reducing operational costs. To hire an expert is costly.
- To replace a retiring or a leaving employee who is an expert.
- Help experts in their routine to improve productivity.
- Help experts in their more complex and difficult tasks
- An expert to obtain information needed by other experts who have forgotten about it or who are too busy to search for it.
- Characteristics of knowledge and inference engines used by domain experts
- Time and money spent on the project.
- Programming Capabilities available in-house.
- Hardware available for development
- Hardware available for deployment
- Required Performance of the system.

Few Applications of Expert system

- ES in Remote sensing applications
- ES in management applications
- ES in Research & Development
- ES for business (Marketing)
- ES for Biologist
- ES for Mathematicians and LAW

Advantages

- Increased productivity (find solutions much faster than humans).
- Availability of expertise (human experts can be at one place at a time).
- Can be used in dangerous environments (e.g. in space).
- Reuse of User Interface and Inference modules
- Level of programming skill needed is low.
- Faster and cheaper completion of project

Limitations

- Difficulty in engineering, especially acquiring the expertise.
- Mistrust by the users.
- Effective only in specific areas (areas of expertise)

The main areas of application of ES are (Waterman, 1986)

Interpretation — drawing high–level conclusions based on data.
Prediction — projecting probable outcomes.
Diagnosis — determining the cause of malfunctions, disease, etc.
Design — finding best configuration based on criteria.
Planning — proposing a series of actions to achieve a goal.
Monitoring — comparing observed behaviour to the expected behaviour.
Debugging and Repair — prescribing and implementing remedies.
Instruction — assisting students in learning.
Control — governing the behaviour of a system.

Steps to Develop an Expert Systems

The process of ES development is iterative. Steps in developing the ES include −

Identify Problem Domain

- The problem must be suitable for an expert system to solve it.
- Find the experts in task domain for the ES project.
- Establish cost-effectiveness of the system.

Design the System

- Identify the ES Technology
- Know and establish the degree of integration with the other systems and databases.
- Realize how the concepts can represent the domain knowledge best.

Develop the Prototype

From Knowledge Base: The knowledge engineer works to −

- Acquire domain knowledge from the expert.
- Represent it in the form of If-THEN-ELSE rules.

Test and Refine the Prototype

- The knowledge engineer uses sample cases to test the prototype for any deficiencies in performance.
- End users test the prototypes of the ES.

Develop and Complete the ES

- Test and ensure the interaction of the ES with all elements of its environment, including end users, databases, and other information systems.
- Document the ES project well
- Train the user to use ES.

Examples of Expert System

Weather forecasting and radar projections use applications of expert systems. These forecasting systems use several rules based on temperature, wind, humidity, and jet stream. The application makes projections on potential weather patterns based on specific conditions and rules within the application.

The logistics management necessary to respond to natural disasters requires applications of expert systems. This software can help emergency personal determine where to place equipment for disaster relief resources. This is most often used in forest firefighting, hurricane relief, and earthquake relief around the world.

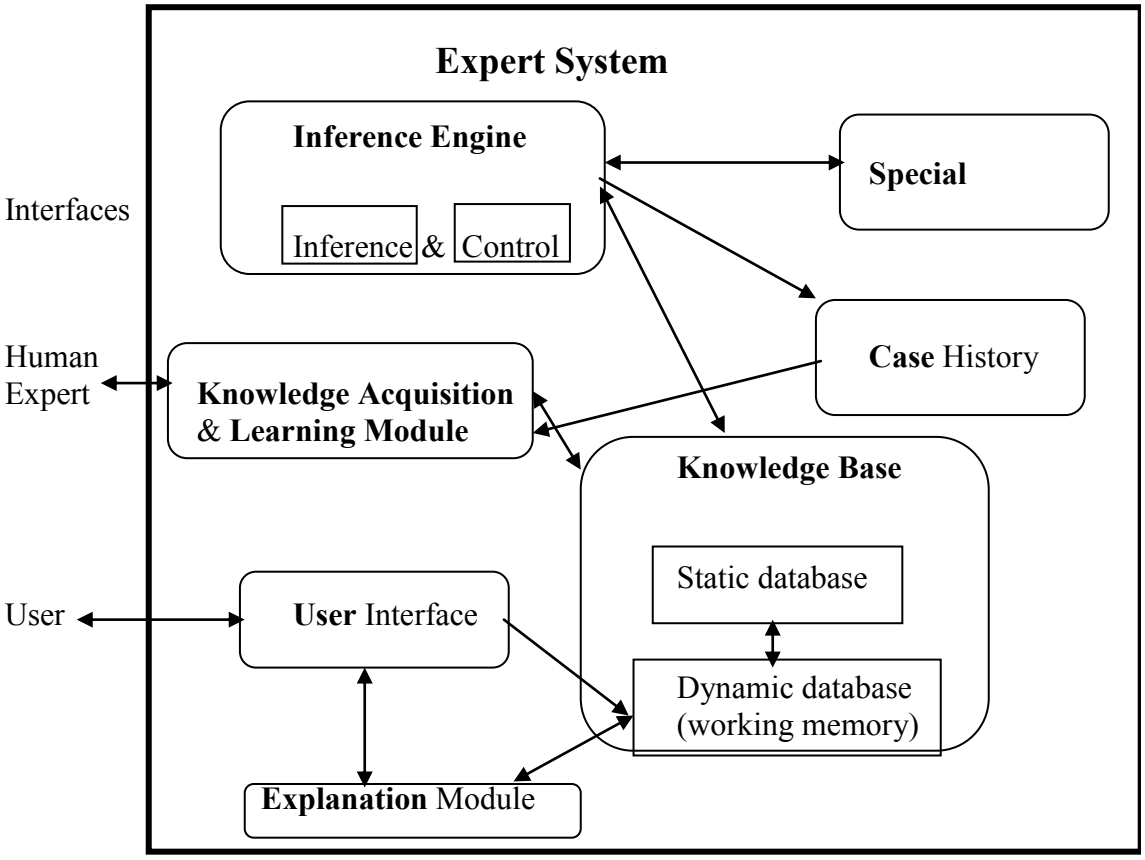
War-simulation programs are applications of expert systems. These programs use complex rules to determine armory logistics and human casualties during warfare situations. The war simulation programs are used to help governments determine the best approach to take during a confrontation.

Human intelligence and experience is what makes an expert system work. These problem-solving techniques are transformed into business rules that help the program weigh options on specific decisions. Each expert system requires human subject matter experts to convert logical process into specific rules.

The human genome project was a program that used expert systems to decipher human genetics. This provided advance mapping features to scientists that helped determine the linkage between genetics and human characteristics. Without expert systems, this linkage would have been difficult to discover.

2. ARCHITECTURE OF EXPERT SYSTEM

..



Knowledge Base:

- Knowledge about problem domain in the form of static and dynamic databases.
- Static knowledge consists of rules and facts which is complied as a part of the system and does not change during execution of the system.
- Dynamic knowledge consists of facts related to a particular consultation of the system.
  - At the beginning of the consultation, the dynamic knowledge base often called working memory is empty.
  - As a consultation progresses, dynamic knowledge base grows and is used along with static knowledge in decision making.
- Working memory is deleted at the end of consultation of the system.

Components of Knowledge Base

The knowledge base of an ES is a store of both, factual and heuristic knowledge.

- **Factual Knowledge** − It is the information widely accepted by the Knowledge Engineers and scholars in the task domain.
- **Heuristic Knowledge** − It is about practice, accurate judgement, one's ability of evaluation, and guessing.

Inference Engine:

- It consists of inference mechanism and control strategy.
- Inference means search through knowledge base and derive new knowledge.
- It involve formal reasoning involving matching and unification similar to the one performed by human expert to solve problems in a specific area of knowledge.
- Inference operates by using modus ponen rule.

- Control strategy determines the order in which rules are applied.
- There are mainly two types of control mechanism viz., forward chaining and backward chaining.

To recommend a solution, the interface engine uses the following strategies −

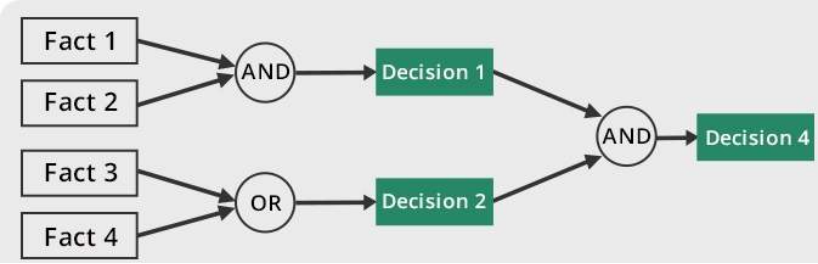
- Forward Chaining
- Backward Chaining

Forward Chaining

It is a strategy of an expert system to answer the question, **"What can happen next?"**

Here, the interface engine follows the chain of conditions and derivations and finally deduces the outcome. It considers all the facts and rules, and sorts them before concluding to a solution.

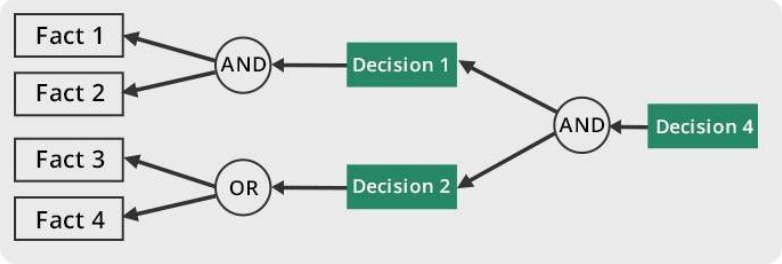
This strategy is followed for working on conclusion, result, or effect. For example, prediction of share market status as an effect of changes in interest rates.



Backward Chaining

With this strategy, an expert system finds out the answer to the question, **"Why this happened?"**

On the basis of what has already happened, the interface engine tries to find out which conditions could have happened in the past for this result. This strategy is followed for finding out cause or reason. For example, diagnosis of blood cancer in humans.



Knowledge Acquisition:

- Knowledge acquisition module allows system to acquire knowledge about the problem domain.
- Sources of Knowledge for ES
  - Text books, reports, case studies,
  - Empirical data and
  - Domain expert experience.
- Updating of Knowledge can be done using knowledge acquisition module of the system.
  - Insertion,
  - Deletion and
  - Updation of existing knowledge

Case History

- Case History stores the file created by inference engine using the dynamic database created at the time of consultation.
- Useful for learning module to enrich its knowledge base.
- Different cases with solutions are stored in Case Base system.
- These cases are used for solving problem using Case Base Reasoning (CBR).

Explanation Module

- Most expert systems have explanation facilities that allow the user to ask the system ***why*** it asked some question, and ***how*** it reached to conclusion.
- It contains 'How' and 'Why' modules attached to it.
  - The sub-module 'How' tells the user about the process through which system has reached to a particular solution
  - 'Why' sub-module tells that why is that particular solution offered.
- It explains user about the reasoning behind any particular problem solution.
- Questions are answered by referring to the system goals, the rules being used, and any existing problem data.

User Interfaces:

Allows user to communicate with system in interactive mode and helps system to create working knowledge for the problem to be solved. The explanation may appear in the following forms −

- Natural language displayed on screen.
- Verbal narrations in natural language.
- Listing of rule numbers displayed on the screen.

Dialogue Module (User Interface)	
System	Do you have fever?
User	Yes
System	Do you have bad throat?
User	No
System	Do you have cough?
User	Yes
System	Are you suffering from running nose?
User	Yes
System	Are you suffering from headache?
User	No

Special Interfaces:

- It may be used for specialized activities such as handling uncertainty in knowledge.
- This is a major area of expert systems research that involves methods for reasoning with uncertain data and uncertain knowledge.
- Knowledge is generally incomplete and uncertain.
- To deal with uncertain knowledge, a rule may have associated with it a *confidence factor* or a weight.
- The set of methods for using uncertain knowledge in combination with uncertain data in the reasoning process is called reasoning with uncertainty.

Types of Expert System Architecture

ES Architecture broadly classified as two types. They are as follows:

1. Production System Architectures (Or) Rule Based System Architectures
2. Non- Production System Architectures
i) Associative/Semantic Network Architectures
ii) Frame Architectures
iii) Decision Tree Architectures
iv) Blackboard System Architectures
v) Analogical Reasoning Architecture
vi) Neural Network Architectures

1. Production System Architecture (Or) Rule Based System

Production systems (or rule–based systems) are programs that instead of conventional algorithms use sets of IF–THEN rules (production rules). Unlike in algorithms, the order in which these rules should be used is not specified. It is decided by the program itself with respect to a problem state. In 1943, Post proved that any computable problem can be implemented in a production system. Cognitive scientists became interested in production systems because they seemed to represent better the way humans think and solve problems.

- Also known as "production systems" or "expert systems"
- Rule-based systems are one of the most successful AI paradigms
- Used for synthesis (construction) type systems
- Also used for analysis (diagnostic or classification) type systems

Label R_n if *condition1*
condition2
…
then
action1
action2 …

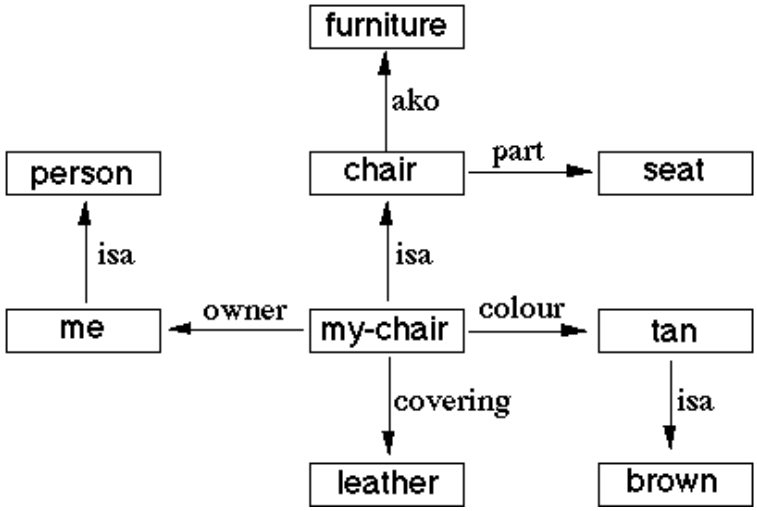
Examples:
IF Saturday OR Sunday THEN go to cinema
IF NOT (Saturday OR Sunday) THEN go to work IF go to cinema THEN go outside
IF go to work AND NOT at work THEN go outside
IF NOT (can go outside) THEN stay home
IF good weather THEN can go outside
IF raining THEN have an umbrella
IF raining AND have an umbrella THEN can go outside

2. Non- Production System Architectures

i) Frame Architecture

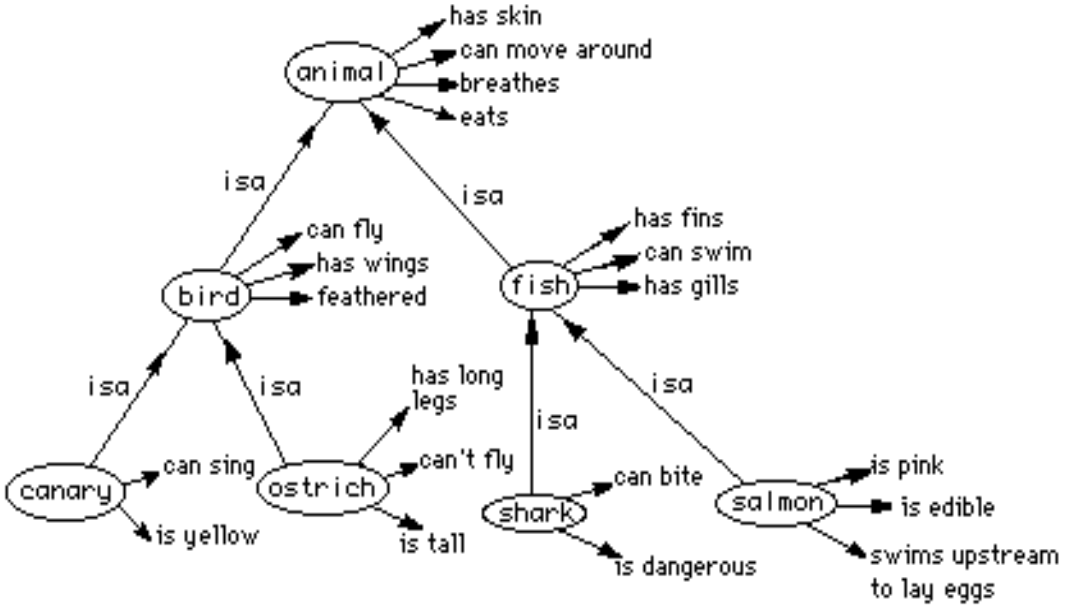
- One type of schema is a frame (or script – time-ordered sequence of frames).
- Frames are useful for simulating commonsense knowledge.
- Semantic nets provide 2-dimensional knowledge; frames provide 3-dimensional.
- Frames represent related knowledge about narrow subjects having much default knowledge
- The attributes, values and procedures are stored in specified slots.
- Slots size may vary.

- Example: PIP (Present Illness Program)
- Medical Expert System.
- Patient findings matched against frames, a trigger status occur.
- A frame is a group of slots and fillers that defines a stereotypical object that is used to represent generic / specific knowledge.
- Commonsense knowledge is knowledge that is generally known.
- Prototypes are objects possessing all typical characteristics of whatever is being modeled.
- Problems with frames include allowing unrestrained alteration / cancellation of slots.



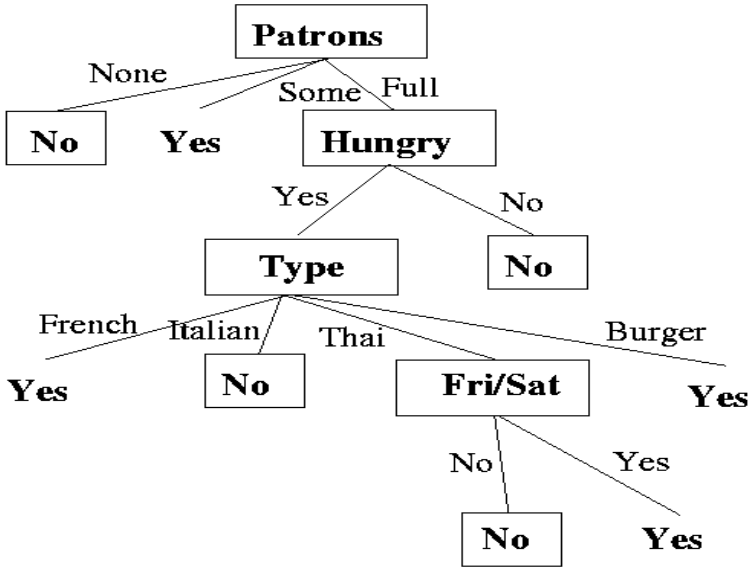
ii) Associative/ Semantic Network Architectures

- A classic representation technique for propositional information
- Propositions – a form of declarative knowledge, stating facts (true/false)
- Propositions are called "atoms" – cannot be further subdivided.
- Semantic nets consist of nodes (objects, concepts, situations) and arcs (relationships between them**).**
- **Common type of links are..**
  IS-A – relates an instance or individual to a generic class
  A-KIND-OF – relates generic nodes to generic nodes



iii) Decision Tree Architecture

- Knowledge for ES may be stored in the form of Decision tree.
- Special tree building editor used.
- New nodes will be added when additional rules are added.
- Traversing technique to identify the attributes.

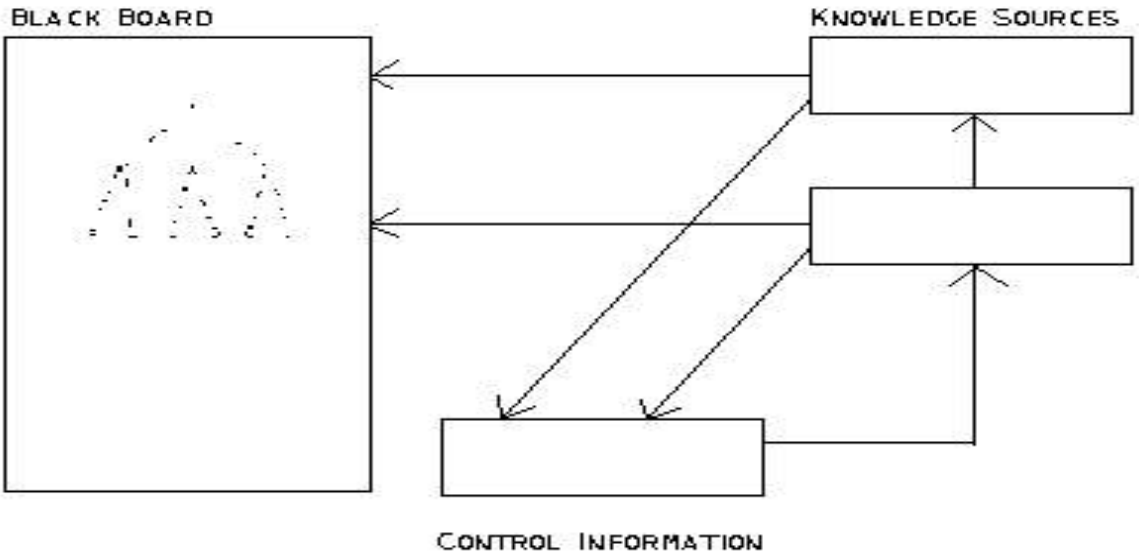


iv) Black Board Architecture

- There are 3 components
- There are number of *knowledge sources*.
- A data structure called *Blackboard* which contains current problem state and information needed by the knowledge sources.
- *Control Information* – Monitors changes in the blackboard.

A blackboard-system application consists of three major components

1. The software specialist modules, which are called **knowledge sources (KSs)**. Like the human experts at a blackboard, each knowledge source provides specific expertise needed by the application.
2. The **blackboard**, a shared repository of problems, partial solutions, suggestions, and contributed information. The blackboard can be thought of as a dynamic "library" of contributions to the current problem that have been recently "published" by other knowledge sources.
3. The **control shell**, which controls the flow of problem-solving activity in the system. Just as the eager human specialists need a moderator to prevent them from trampling each other in a mad dash to grab the chalk, KSs need a mechanism to organize their use in the most effective and coherent fashion. In a blackboard system, this is provided by the control shell.

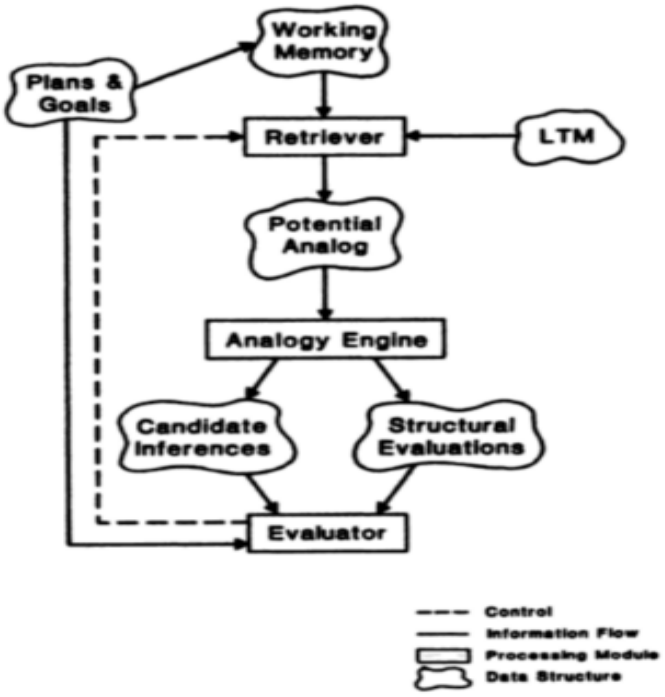


v) Analogical Reasoning Architecture

Analogy-based reasoning: This term is sometimes used, as a synonym to case-based reasoning, to describe the typical case-based approach... However, it is also often used to characterize methods that solve new problems based on past cases from a different domain, while typical case-based methods focus on indexing and matching strategies for single-domain cases."

- Expert System is based on analogical structure
- Applying known solution to unknown problems.
- Example:
  Product of odd numbers is odd-Known
  Find Product of even numbers?

Architecture:



3. ROLES OF EXPERT SYSTEM

Three fundamental roles in building expert systems are:
1. *Expert* - Successful ES systems depend on the experience and application of knowledge that the people can bring to it during its development. Large systems generally require multiple experts.

2. *Knowledge engineer* - The knowledge engineer has a dual task. This person should be able to elicit knowledge from the expert, gradually gaining an understanding of an area of expertise. Intelligence, tact, empathy, and proficiency in specific techniques of knowledge acquisition are all required of a knowledge engineer. Knowledge-acquisition techniques include conducting interviews with varying degrees of structure, protocol analysis, observation of experts at work, and analysis of cases.
On the other hand, the knowledge engineer must also select a tool appropriate for the project and use it to represent the knowledge with the application of the ***knowledge acquisition facility***.

3. *User* - A system developed by an end user with a simple shell, is built rather quickly an inexpensively. Larger systems are built in an organized development effort. A prototype-oriented iterative development strategy is commonly used. ESs lends themselves particularly well to prototyping. In artificial intelligence, an **expert system** is a computer system that emulates the decision-making ability of a human expert. Expert systems are designed to solve complex problems by reasoning about knowledge, represented primarily as if–then rules rather than through conventional procedural code

ES covers the following list of problem types in ES applications.

Interpretation: An interpretation system takes observed data and explains its meaning by inferring the problem state which corresponds to the observed data. Examples are Dipmeter Advisor, a system for interpreting geophysical oil well log data, and Prospector, a system for identifying geological ore-bearing formations.

Diagnosis: Diagnosis systems infer malfunctions or system state from observed irregularities and interpretation of data. MYCIN, an infectious disease diagnostician and several other medical diagnosis programs fall into this category.

Monitoring: A monitor observes system behavior and compares the observations to the planned behavior to determine flaws in the plan or potential malfunctions of the system. An example is Ventilation Manager, a program for monitoring a patient's ventilation therapy.

Design: Design is the process of developing a configuration for an object which satisfies all applicable constraints. R1 (or XCON) is an example of a design system which is used to configure VAX4 computers.

Planning: Planning is a design process that yields a set a actions intended to produce a desired outcome. An example is MOLGEN, a KBES for planning experiments in molecular genetics.

Control: A control system encompasses many of the characteristics of the other types of applications described above. It must interpret data, predict outcomes, formulate plans, execute the plans, and monitor their execution.

Critics: There is a great potential for a special class of diagnostic KBES, which may be called critics. The role of these KBES can be explained as follows. Generative KBES such as HIRISE and CARTER use reformulated constraints to guide the search for feasible solutions. However, there are many "soft" constraints that may affect the feasibility or optimality of a candidate design or plan which are not known with sufficient precision to be formulated as prior constraints. These soft constraints are not in the knowledge base of the designers, but are typically part of the expertise of the constructors, manufacturers or users of the system or product being designed. These experts, in turn, cannot be expected to tell the designer how to design something so that their constraints are satisfied; all they can be expected to do is to evaluate or critique a proposed candidate design and assert whether their constraints are satisfied or not. Presumably the designer, when presented with such a critique, can modify his design so as to eliminate any cause of negative criticism

Generators: The key characteristics of such systems are:
• The constraints are represented in the knowledge base;
• The set of known entities, or rules for generating them, are in the knowledge base; therefore, these systems will not "invent" new plans or designs, but, may still generate novel candidate solutions by arranging or combining the known entities in unexpected ways.
• While the choice of inference strategy is not as clear as for diagnostic KBES, a number of inference strategies have been proposed or evaluated and can serve as a prototypes. The "best" candidate solution is understood to mean best among candidates that can be generated from the known entities, measured according to some evaluation function. Again, the knowledge base can grow by the addition of new candidate entities, new constraints or new rules for combining entities.

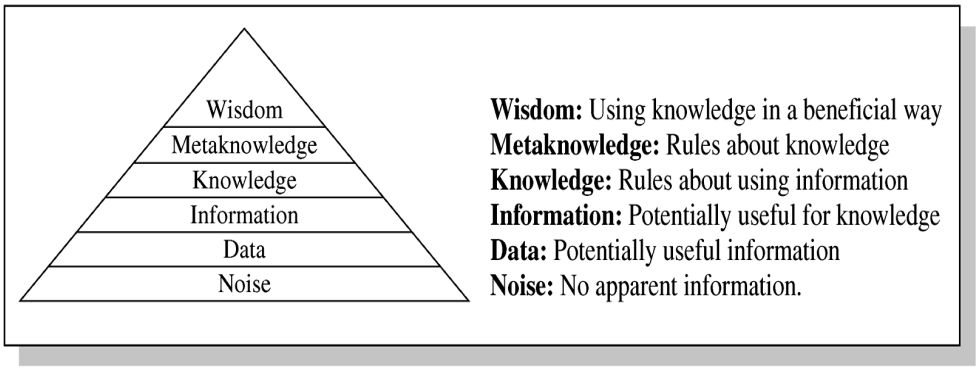
4. KNOWLEDGE ACQUISITION

Basic Concepts

- Knowledge is power, often inexact & incomplete and often poorly specified
- Amateurs become experts slowly
- Expert systems must be flexible and transparent
- Separate inference engine and knowledge base (make system easy to modify)
- Use uniform "fact" representation (reduces number of rules required and limits combinatorial explosion)
- Keep inference engine simple (makes knowledge acquisition and truth maintenance easier)
- Exploit redundancy (can help overcome problems due to inexact or uncertain reasoning)

Meta Knowledge

- Meta knowledge is knowledge about knowledge and expertise.
- Most successful expert systems are restricted to as small a domain as possible.
- In an expert system, ontology is the Meta knowledge that describes everything known about the problem domain.
- Wisdom is the Meta knowledge of determining the best goals of life and how to obtain them.



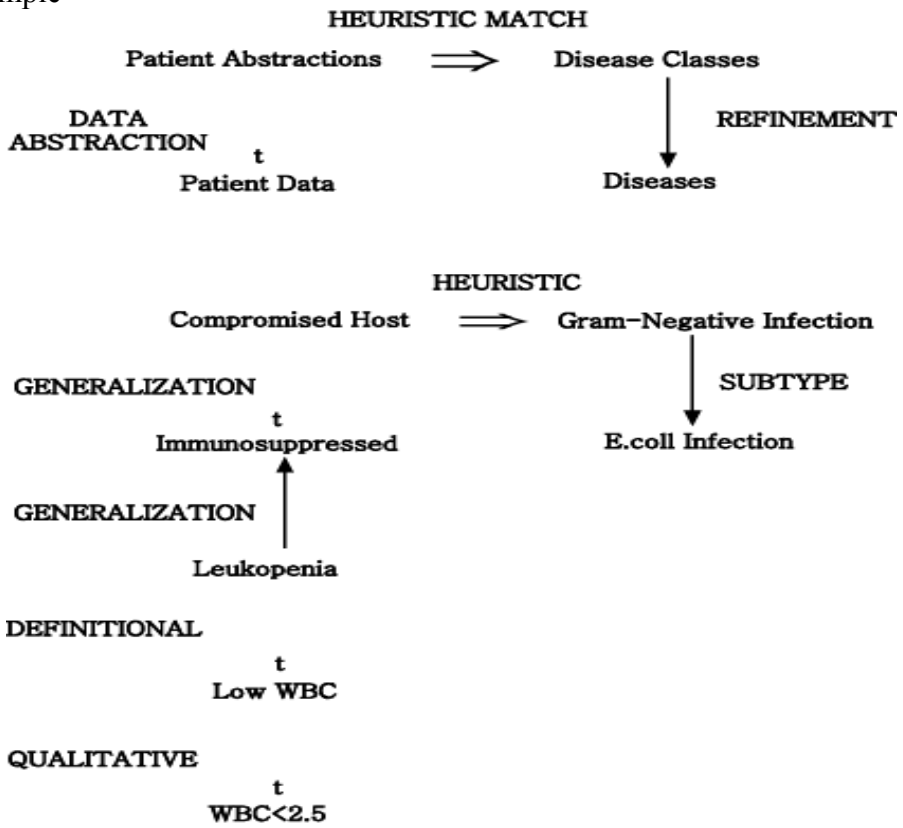
Pyramid of Knowledge

Heuristics

Expert systems apply heuristics to guide the reasoning and thus reduce the search area for a solution. It is about practice, accurate judgment, one's ability of evaluation, and guessing.

Heuristic classification

In simple classification, data may directly match solution features or may match after being abstracted. In heuristic classification, solutions and solution features may also be matched heuristically, by direct, non-hierarchical association with some concept in another classification hierarchy. For example, MYCIN does more than identify an unknown organism in terms of visible features of an organism: MYCIN heuristically relates an abstract characterization of the patient to a classification of diseases. We show this inference structure schematically, followed by an example

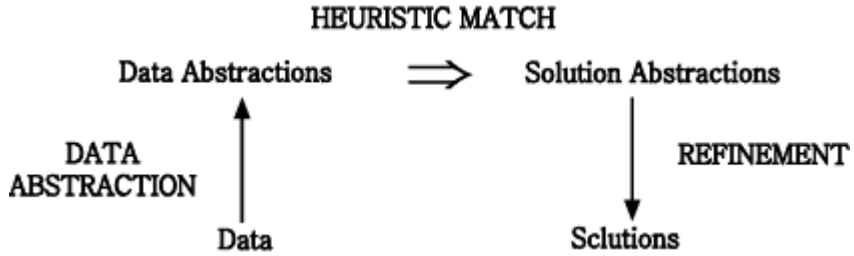


Inference structure of MYCIN

Basic observations about the patient are abstracted to patient categories, which are heuristically linked to diseases and disease categories. While only a subtype link with E.coli infection is shown here, evidence may actually derive from a combination of inferences. Some data might directly match E.coli features (an individual organism shaped like a rod and producing a Gram-negative stain is seen growing in a culture taken from the patient). Descriptions of laboratory cultures (describing location, method of collection, and incubation) can also be related to the classification of diseases.

The important link we have added is a heuristic association between a characterization of the patient (¡°compromised host¡±) and categories of diseases (¡°gram-negative infection¡±). Unlike definitional and hierarchical inferences, this inference makes a great leap. A heuristic relation is uncertain, based on assumptions of typicality, and is sometimes just a poorly understood correlation. A heuristic is often empirical, deriving from problem-solving experience; heuristics correspond to the ¡°rules of thumb,¡± often associated with expert systems (Feigenbaum, 1977).

Heuristics of this type reduce search by skipping over intermediate relations. These associations are usually uncertain because the intermediate relations may not hold in the specific case. Intermediate relations may be omitted because they are unobservable or poorly understood. In a medical diagnosis program, heuristics typically skip over the causal relations between symptoms and diseases. To summarize, in heuristic classification abstracted data statements are associated with specific problem solutions or features that characterize a solution. This can be shown schematically in simple terms.



Inference structure of heuristic classification

This diagram summarizes how a distinguished set of terms (data, data abstractions, solution abstractions, and solutions) are related systematically by different kinds of relations. This is the structure of inference in heuristic classification.

Criteria of Selecting Problem

- Recognized experts exist
- Experts do better than amateur
- Expert needs significant time to solve it
- Cognitive type tasks
- Skill can routinely taught to beginners
- Domain has high payoff
- Task does not require common sense

Knowledge Acquisition Process

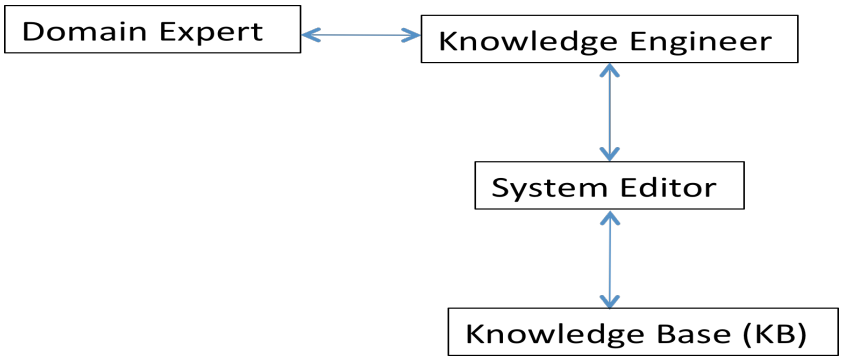
What is Knowledge Acquisition?

Knowledge acquisition is transferring knowledge from human expert to computer. Knowledge acquisition includes the elicitation, collection, analysis, modeling and validation of knowledge.

- Process is similar to rapid prototyping (expert is the customer)
- Expert is involved throughout the development process
- Incremental systems are presented to expert for feedback and approval
- Change is viewed as healthy not a process failure
- Expert may not have required knowledge in some areas
- Expert may not be consciously aware of required knowledge needed
- Expert may not be able to communicate the knowledge needed to knowledge engineer
- Knowledge engineer may not be able to structure knowledge for entry into knowledge base.

Steps in knowledge acquisition

1. Collect: (elicitation)
   - Getting the knowledge out of the expert
   - Most difficult step
   - Lots of strategies
2. Interpret:
   - Review collected knowledge, organize, and filter
3. Analyze:
   - Determining types of knowledge, conceptual relationships
   - Determining appropriate knowledge representations & inference structure
4. Design:
   - Extracting more knowledge after using above principles



Domain Expert

- Anyone can be considered a **domain expert** if he or she has deep knowledge (of both facts and rules) and strong practical experience in a particular domain. The area of the domain may be limited. In general, an expert is a skilful person who can do things other people cannot.Eg: Doctor
- Provides knowledge and processes needed to solve problem
- Domain expert must be available for hundreds of hours
- Knowledge in the expert system ends up being the knowledge engineer's understanding of the domain, not the domain expert's knowledge

Knowledge Engineer

Knowledge engineers are involved with validation and verification.

Validation is the process of ensuring that something is correct or conforms to a certain standard. A knowledge engineer is required to carry out data collection and data entry, but they must use validation in order to ensure that the data they collect, and then enter into their systems, fall within the accepted boundaries of the application collecting the data.

It is important that a knowledge engineer incorporates validation procedures into their systems within the program code. After the knowledge-based system is constructed, it can be maintained by the domain expert

System Editor

- Knowledge base editor which help the expert or knowledge engineer to easily update and check the knowledge base .

Knowledge Base:

- Knowledge about problem domain in the form of static and dynamic databases.
- Static knowledge consists of rules and facts which is complied as a part of the system and does not change during execution of the system.
- Dynamic knowledge consists of facts related to a particular consultation of the system.
  - At the beginning of the consultation, the dynamic knowledge base often called working memory is empty.
  - As a consultation progresses, dynamic knowledge base grows and is used along with static knowledge in decision making.
- Working memory is deleted at the end of consultation of the system.

Components of Knowledge Base

The knowledge base of an ES is a store of both, factual and heuristic knowledge.

- **Factual Knowledge** − It is the information widely accepted by the Knowledge Engineers and scholars in the task domain.
- **Heuristic Knowledge** − Expert systems apply heuristics to guide the reasoning and thus reduce the search area for a solution. It is about practice, accurate judgment, one's ability of evaluation, and guessing.

Truth Maintenance

- Task of maintaining the logical consistency of the rules in the rule-base
- Given the incremental manner in which rule-bases are built and since rules themselves are modular their interactions are hard to predict
- Newly added rules can render old rules obsolete and can be inconsistent with existing rules

  1. Knowledge about components, e.g. voltage, ampere, priority, no of ports and so on.
  2. Knowledge about constraints i.e. Rules for forming partial configuration of equipment's and extending successfully

Knowledge Acquisition Difficulties:

- Expert may not have required knowledge in some areas
- Expert may not be consciously aware of required knowledge needed
- Expert may not be able to communicate the knowledge needed to knowledge engineer
  Knowledge engineer may not be able to structure knowledge for entry into knowledge base

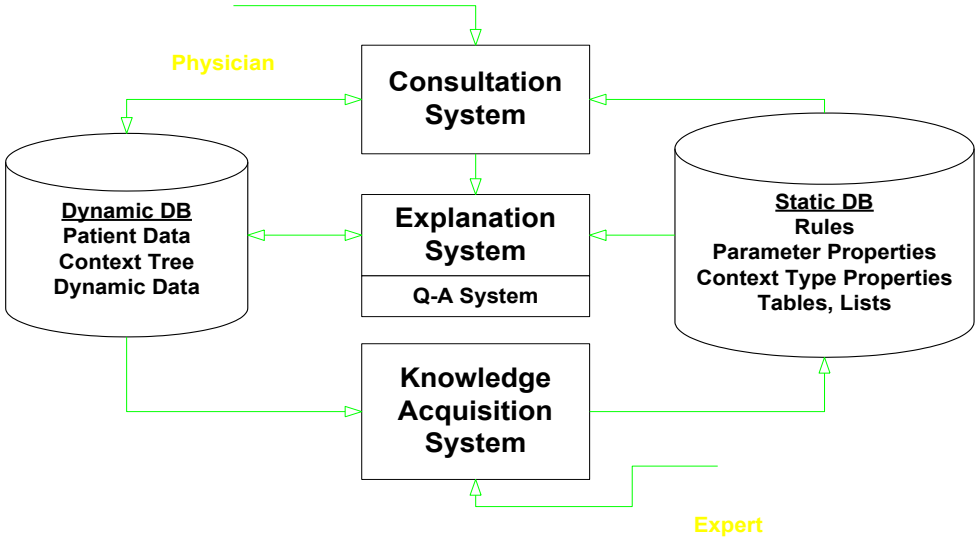
5. TYPICAL EXPERT SYSTEMS

i) MYCIN

Mycin operated using a fairly simple inference engine, a knowledge base of ~500 rules. It would query the physician running the program via a long series of simple yes/no or textual questions. At the end, it provided a list of possible culprit bacteria ranked from high to low based on the probability of each diagnosis, its confidence in each diagnosis' probability, the reasoning behind each diagnosis (that is, Mycin would also list the questions and rules which led it to rank a diagnosis a particular way), and its recommended course of drug treatment.

Despite Mycin's success, it is sometimes now used as a cautionary tale in AI courses for people who generate their own ad hoc probability frameworks. Mycin had a limited depth to its reasoning hierarchy due to the noise introduced by its certainty factor system. This problem can be solved by using a rigorous probabilistic framework, such as Bayesian statistics.

Architecture:



- Performs Diagnosis and Therapy Selection
- Control Structure reads Static DB (rules) and read/writes to Dynamic DB (patient, context)
- Linked to Explanations
- Terminal interface to Physician

Static Database

- Rules
- Meta-Rules
- Templates
- Rule Properties
- Context Properties
- Fed from Knowledge Acquisition System

Dynamic Database

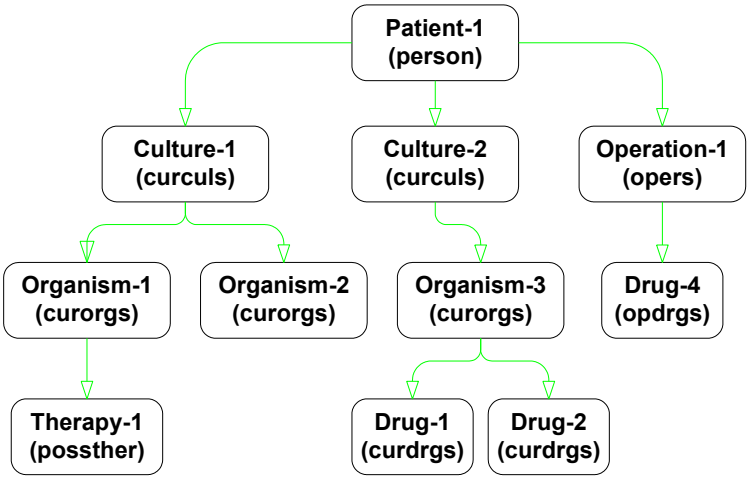
- Rules
- Meta-Rules

- Templates
- Rule Properties
- Context Properties
- Fed from Knowledge Acquisition System

Explanation System

- Provides reasoning why a conclusion has been made, or why a question is being asked
- Q-A Module
- Reasoning Status Checker
- Extends Static DB via Dialogue with Experts
- Dialogue Driven by System
- Requires minimal training for Experts

Context Tree



ii) DART

- In anticipation of computer faults, most manufacturers prepare specialized diagnostic aids that allow field engineers without complete knowledge of a system to diagnose the majority of its failures. Unfortunately, these diagnostics are not infallible for they do not take into account every fault and combination of faults for every possible system configuration. Thus, in some cases, it is necessary to call in someone with expert knowledge about the design of the system. These experts are expensive and often not immediately available, and there is inevitably a delay and a loss of working time to the system users.
- DART is a joint project of the Heuristic Programming Project and IBM that explores the application of artificial intelligence techniques to the diagnosis of computer faults. The primary goal of the DART Project is to develop programs that capture the special design knowledge and diagnostic abilities of these experts and to make them available to field engineers. The practical goal is the construction of an automated diagnostician capable of pinpointing the functional units responsible for observed malfunctions in arbitrary system configurations.
-

iii) XOON

- Expert system to configure VAX -11/780 computers
- Developed by collaboration between Carnegie Mellon University and Digital Equipment Corporation
- Data driven approach

iv) EXPERT SYSTEM SHELL

Expert system shells are the software containing an interface, an inference engine, and the formatted skeleton of a knowledge base. In essence, an **expert system shell** is an empty bowl to be filled with the **expert** knowledge elements that the inference engine may process for users.

The E.S shell simplifies the process of creating a knowledge base. It is the shell that actually processes the information entered by a user relates it to the concepts contained in the knowledge base and provides an assessment or solution for a particular problem. Thus E.S shell provides a layer between the user interface and the computer O.S to manage the input and output of the data. It also manipulates the information provided by the user in conjunction with the knowledge base to arrive at a particular conclusion.

Knowledge Representation

Expert systems shells provide the bare bones for imitation of human expert reasoning in rule methods known as forward chaining and backward chaining.

Forward chaining in these shells enables taking data from a user and using inference engine rules to locate more data relative to that information until there is enough information to form a conclusion. Because the initial data received is what drives the seeking, this method is called a data driven method. An application that illustrates this forward-chaining method might explore the possibilities of arrangement of components within a computer to arrive at the best placement of the components.

Backward chaining gathers data only as it needs it when a knowledge base is being queried on a consultation. It has the goal of finding a value for C and reasons backward to discover the value of A and B that conclude the goal value of C. This method of reasoning from present data to prior data that was the underpinning of present data is called goal-driven method. An application illustrating expert system shells rules of inference might include a medical doctor inputting a current set of symptoms for background information on the same or similar symptoms in background information from a particular medical diagnosis expert system.

Inferred knowledge is gained by examination of existing facts to arrive at likely new information. This is the reasoning process that inhabits the inference engine in expert system shells.