CHAPTER 6

Unit VI

Expert System

Syllabus

6.1 Hybrid Approach - Fuzzy Neural Systems

6.2 Expert system : Introduction, Characteristics, Architecture, Stages in the development of expert system,

6.1 Hybrid Approach

6.1.1 Introduction to Hybrid Systems

- Hybrid systems are those for which more than one soft computing technique is integrated to solve a real-world problem.
- Neural networks, fuzzy logic and genetic algorithms are soft computing techniques which have been inspired by biological computational processes.
- Each of these technologies has its own advantages and limitations. For example, while neural networks are good at recognizing patterns, they are not good at explaining how they reach their decisions.
- Fuzzy logic systems, which can reason with imprecise information, are good at explaining their decisions but they cannot automatically acquire the rules they use to make those decisions.
- These limitations have been a central driving force behind the creation of intelligent hybrid systems where two or more techniques are combined in a manner that overcomes the limitations of individual techniques.

6.1.2 Types of Hybrid Systems

Hybrid systems can be classified as:

1. Sequential hybrids
2. Auxiliary hybrids
3. Embedded hybrids

1. Sequential hybrid systems

- In sequential hybrid systems, all the technologies are used in a pipe-line fashion. The output of one technology becomes the input to another technology (Fig. 6.1.1).
- This kind of hybridization form is one of its weakest, because it does not integrate different technologies into a single unit.

- An example is a GA pre-processor which obtains the optimal parameters such as initial weights, threshold, learning rate etc. and hands over these parameters to a neural network.

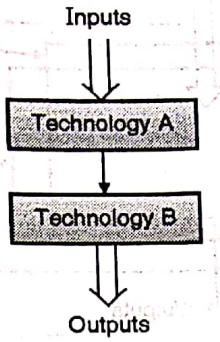

Fig.6.1.1 : A sequential hybrid system

2. Auxiliary hybrid systems

- In this type, a technology treats another technology as a "subroutine" and calls it to process or generate whatever information is needed by it. Fig. 6.1.2 illustrates the auxiliary hybrid system.

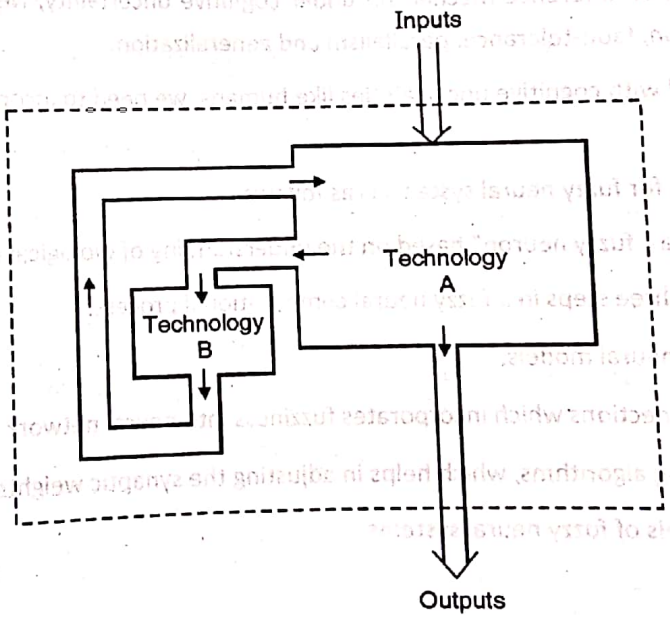

Fig. 6.1.2 : An auxiliary system

- An example is a neuro - genetic system in which an NN employs a GA to optimize its structural parameters, i.e. parameters which define Neural Network's architecture.

3. Embedded hybrid systems

- In embedded hybrid systems, the technologies are integrated in such a manner that they appear to be intertwined.
- The fusion is so complete that it would appear that no technology can be used without the other for solving the problem. Fig. 6.1.3 depicts the schema for an embedded hybrid system.
- Example of this system is a NN-FL (Neural Network Fuzzy Logic) hybrid system that has NN which receives information, processes it and generates fuzzy outputs as well.

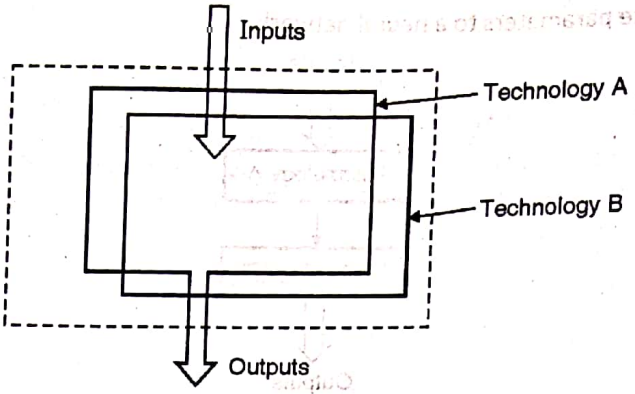

Fig. 6.1.3 : An embedded system

6.1.3 Fuzzy Neural System

- Fuzzy neural system is a system with seamless integration of fuzzy logic and neural networks.
- While fuzzy logic provides an inference mechanism under cognitive uncertainty, Neural networks offer advantages, such as learning, adaptation, fault-tolerance, parallelism and generalization.
- To enable a system to deal with cognitive uncertainties like humans, we need to incorporate the concept of fuzzy logic into the neural networks.
- The computational process for fuzzy neural systems is as follows.
- The first step is to develop a " fuzzy neuron" based on the understanding of biological neuronal morphologies.
- This leads to the following three steps in a fuzzy neural computational process.
  1. Development of fuzzy neural models.
  2. Models of synaptic connections which incorporates fuzziness into neural network
  3. Development of learning algorithms, which helps in adjusting the synaptic weights.
- There are two possible models of fuzzy neural systems:

(a) Model 1 :

- In this model the output of fuzzy system is fed as an input to the neural networks.
- The input to the system is linguistic statements. The fuzzy interface block converts these linguistic statements into an input vector which is then fed to a multi-layer neural network. The neural network can be adapted (trained) to yield desired command outputs or decisions.

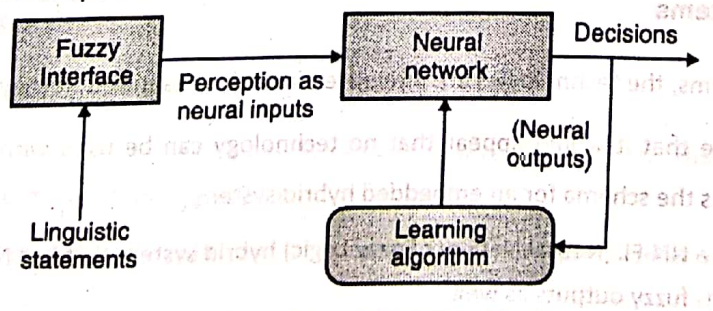

Fig 6.1.4 Fuzzy Neural System : Model 1

(b) Model 2 :

- In second model, a multi-layered neural network drives the fuzzy inference mechanism. That is the output of neural networks is fed as an input to the fuzzy system.

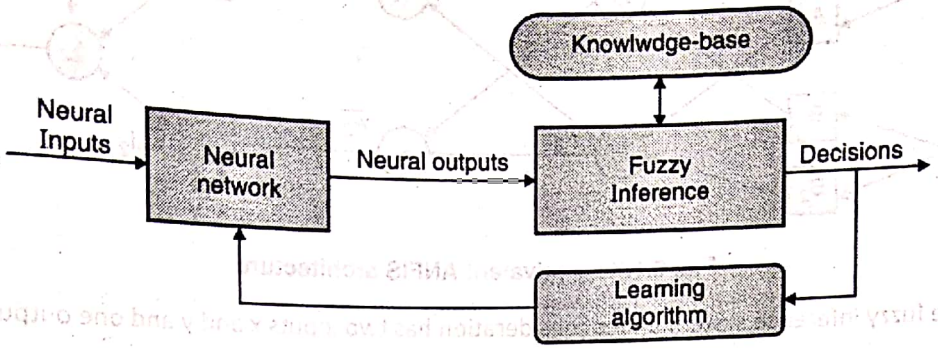

Fig 6.1.5 Fuzzy Neural System : Model 2

- Here, neural networks are used to tune membership functions of fuzzy systems that are employed as decision-making systems for controlling equipment. Neural network learning techniques can automate this process and substantially reduce development time and cost while improving performance.

6.1.4 Adaptive Neuro - Fuzzy Inference System (ANFIS)

MU – Dec. 14, Dec. 15

Q. Explain the architecture of ANFIS with the help of a diagram. (Dec. 14, 10 Marks)

Q. Draw the five layer architecture of ANFIS and explain each layer in brief. (Dec. 15, 5 Marks)

- A well –known and practically used Neuro-Fuzzy system is ANFIS (Adaptive Neuro –fuzzy Inference system) explained below. It is based on Takagi–Sugeno fuzzy inference system developed in 1990s.
- ANFIS is a neural network which is functionally equivalent to a fuzzy inference model.
- Since it integrates both neural networks and fuzzy logic, it has the potential to capture the benefits of both in a single framework.

ANFIS Architecture:

- Fig. 6.1.6 shows the first-order Sugeno model and its equivalent ANFIS architecture is shown in Fig. 6.1.7.

$$f_1 = p_1 x + q_1 y + r_1$$

$$\Rightarrow f = \frac{w_1 f_1 + w_2 f_2}{w_1 + w_2} = \bar{w}_1 f_1 + \bar{w}_2 f_2$$

$$f_2 = p_2 x + q_2 y + r_2$$

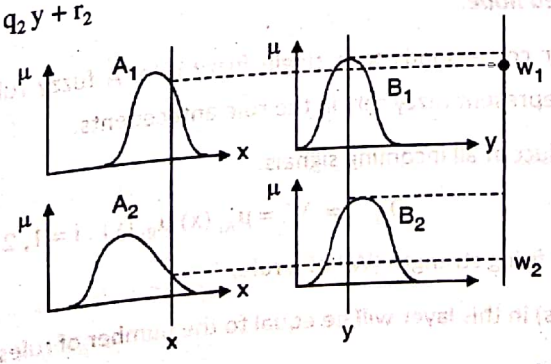

Fig. 6.1.6 : A two-input first-order Sugeno model

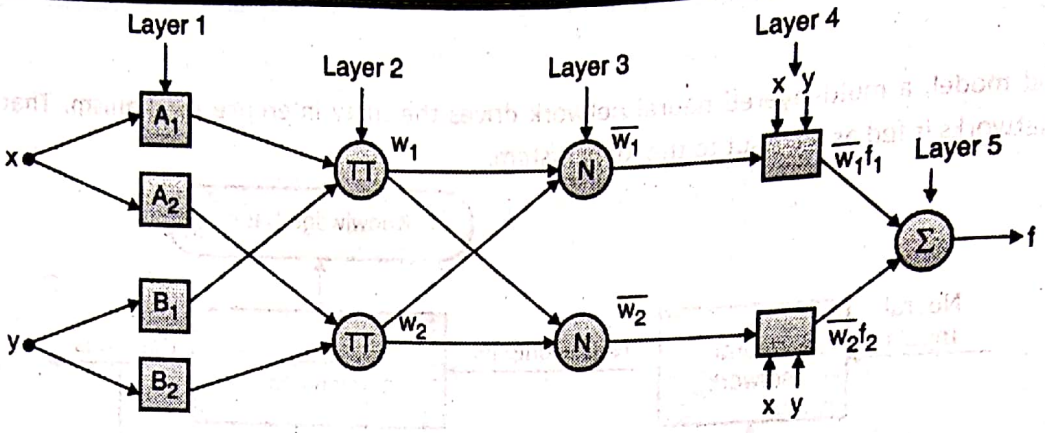

Fig. 6.1.7 : Equivalent ANFIS architecture

- Assume that the fuzzy inference system under consideration has two inputs x and y and one output.
- For a first-order Sugeno fuzzy model, a common rule set with two fuzzy rules is,

  Rule 1: If x is A_1 and y is B_1 then $f_1 = p_1x + q_1y + r_1$

  Rule 2: If x is A_2 and y is B_2 then $f_2 = p_2x + q_2y + r_2$
- The architecture contains five layers.

Layer 1 : Fuzzification layer

- Neurons in this layer represent fuzzy sets used in the antecedents of fuzzy rules. A neuron in the fuzzification layer receives a crisp input and determines the degree to which this input belongs to the neuron's fuzzy set.
- Every node in this layer is an ***adaptive node*** and a node function can be represented as,

$$O_{1,i} = \mu_{Ai}(x), \text{ for } i = 1, 2 \text{ or}$$

$$O_{1,j} = \mu_{Bi-2}(y), \text{ for } i = 3, 4$$

  Where x (or y) are the input to the node i and A_i (or B_{i-2}) is a linguistic label associated with this node.
- Any appropriate membership function can be used such as generalized bell function

$$\mu_A(x) = \frac{1}{1 + \left| \frac{x - c_i}{a_i} \right|^{2b}}$$

  where, $\{a_i, b_i, c_i\}$ is the parameter set. Parameters in this layer are called **premise parameters.**

Layer 2 : Fuzzy rule layer

- Every node in this layer is the ***fixed node***.
- Each node (neuron) in this layer corresponds to a single fuzzy rule. A fuzzy rule neuron receives inputs from the fuzzification layer neurons that represent fuzzy sets in the rule antecedents.
- The output of i^{th} node is the product of all incoming signals.

$$O_{2i} = W_i = \mu_{Ai}(x)\, \mu_{Bi}(y), \; i = 1, 2$$

- Each node output represents the firing strength (W_i) of a rule.
- The number of neurons (or nodes) in this layer will be equal to the number of rules.

Layer 3 : Normalization layer

- Every node in this layer is a ***fixed node***. The i^{th} node of the layer calculates the ratio of the i^{th} rule's firing strength to the sum of all rules' firing strengths.

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}, \; i = 1, 2$$

- The output of this layer is the normalized firing strength.

Layer 4 : Output membership layer

- Nodes in this layer represent fuzzy sets used in the consequent of fuzzy rules.
- Every node in this layer is an ***adaptive node*** with a node function represented as,

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i), \quad i=1, 2$$

where w_i is the normalized firing strength from layer 3 and $\{p_i, q_i, r_i\}$ is the parameter set of this node.

- Parameters in this layer are referred to as **consequent parameters**.

Layer 5 : Defuzzification layer

- Node in this layer is a ***fixed node***.
- Every node in this layer represents a single output of the neuro-fuzzy system. It takes the output fuzzy sets clipped by the respective integrated firing strengths and combines them into a single fuzzy set.
- Output of this node can be represented as,

$$O_{5,1} = \Sigma \bar{w}_i f_i = \frac{\Sigma_i w_i f_i}{\Sigma_i w_i}, \quad i = 1, 2$$

- The structure of this adaptive network is not unique. The alternative structure is shown Fig. 6.1.8.

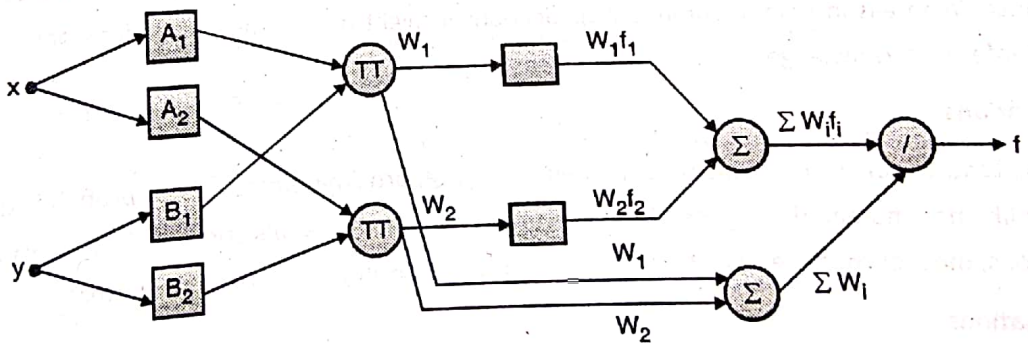

Fig. 6.1.8 : Alternative ANFIS architecture for Sugeno fuzzy model

- Here weight normalization is performed in the last layer.

6.2 Expert System

6.2.1 Introduction

- Jackson (1999) has defined Expert Systems as follows:
- "An expert system is a computer program that represents and reasons with knowledge of some specialist subject with a view to solve problems or give advice."

- Artificial intelligent aims to implement intelligence in machines by developing computer programs that exhibit intelligent behavior. To develop systems which are capable of solving complex problems, it needs efficient access to substantial domain knowledge, a good reasoning mechanism and an effective and efficient way of representing knowledge and inferences; to apply the knowledge to the problems they are supposed to solve. Expert systems also need to explain to the users how they have reached their decisions.
- Expert systems are generally based on the way of knowledge representation, production rule formation, searching mechanism and so on. Usually, to build an expert system, system's shell is used, which is an existing knowledge independent framework, into which domain knowledge can be added to produce a working expert system. This avoids programming for each new system from scratch.
- Often, the two terms, Expert Systems (ES) and Intelligent Knowledge Based Systems (IKBS), are used synonymously. Expert systems are programs whose knowledge base contains the knowledge used by human experts, in place of knowledge gathered from textbooks or non-experts. Expert systems have the most widespread areas of artificial intelligence application.

6.2.2 Characteristics of Expert Systems

Following characteristics distinguish expert systems from conventional computer applications.

1. **Simulation of human reasoning**

   Expert systems simulate human reasoning process in the given problem domain, whereas computer applications try to simulate the domain itself.

2. **Representation of human knowledge**

   Expert systems use various methods to represent the domain knowledge gathered from human expert. It performs reasoning over representations of human knowledge, in addition to numerical calculations or data retrieval. In order to do this, expert systems have corresponding distinct modules referred to as the **inference engine** and the **knowledge base**. Whereas in case of computer applications it might be just the calculations performed on available data, without inference knowledge.

3. **Use approximations**

   Expert systems tend to solve the problems using heuristics or approximate methods or probabilistic methods which are very much like how human do in general. While in case of computer applications strict algorithms are follows to produce solutions, most of the times which do not guarantee the result to be correct or optimal.

4. **Provide explanations**

   Expert systems usually need to provide **explanations** and **justifications** of their solutions or recommendations in order to make user understand their reasoning process to produce a particular solution. This type of behavior is hardy observed in case of computer applications.

Some typical tasks performed by existing expert system are as follows:

- **Data interpretation :** There are different types of data to be interpreted by expert system, which have various formats and features. Example: sonar data, geophysical measurements.
- **Diagnosis of malfunctions :** While collecting data from machines or from experts there can be shortfall of accuracy or mistakes in readings. Example equipment faults or human diseases.

- **Structural analysis :** If system is build for a domain where complex objects like, chemical compounds, computer systems are used; configuration of these complex objects must be studied by the expert system.
- **Planning :** Expert systems are required to plan sequences of actions in order to perform some task. Example: actions that might be performed by robots.
- **Prediction :** Expert systems need to predict the future basis on past knowledge and current information available. Example: weather forecast, exchange rates, share prices, etc.
- **High Performance :** Expert systems are generally preferred because of their high performance. In the sense, they can process huge amount of data to produce results considering several details, in acceptable amount of time. The response time is very less.
- **Highly responsive :** Expert system are required to be highly responsive and user friendly. User can ask any query system should be able to produce the appropriate reply to it. Even if the query asked by user is not answerable with the existing knowledgebase, expert system should give some informative reply about the question.
- **Reliable :** Expert systems are highly reliable as they process huge amount of database. Hence the results produce by the system are always close to exact.

6.2.3 Real Time Expert Systems

There are no fundamental limits on what problem domains an expert system can be built to deal with. Expert systems can be developed almost for every domain for which requires human expert. However, the domain should ideally be one in which an expert requires a few hours to accomplish the task. This section explains some of the expert systems which have been created for such domains.

(i) **Dendral :** This system is considered to be the first expert system in existence. Dendral identifies the molecular structure of unknown compounds. It was developed by Stanford University.

(ii) **Mycin :** This is a milestone in expert system development which, made significant contributions to the medical field; but was not used in actual practice. It provides assistance to physicians in the diagnosis and treatment of meningitis and bacterial infections. It was developed by Stanford University.

(iii) **Altrex :** It helps to diagnose engine troubles of certain models of Toyota cars, used in a central service department which can be called up by those actually servicing the cars for assistance, if required. It was developed by their research lab.

(iv) **Prospector :** This expert system is successful to locate deposits of several minerals, including copper and uranium. It was developed by SRI International.

(v) **Predicate :** This expert system provides estimate of the time required to construct high rise buildings. It was developed by Digital Equipment Corporation for use by Land Lease, Australia.

6.3 Building Blocks of Expert Systems

MU -Dec. 15, May 16

Q. Draw and describe the architecture of expert system. **(Dec.15, 6 Marks)**

Q. Draw and explain architecture of expert system **(May 16, 5 Marks)**

- Let's first understand the components of expert systems.

- Following are the two principal components of every expert system.

1. Knowledge base
2. The reasoning or inference engine

- Every expert system is developed for a specific task domain. It is the area where human intelligence is required. Task refers to some goal oriented problem solving activity and Domain refers to the area within which the task is being performed.
- Consider the expert system architecture in Fig. 6.3.1.

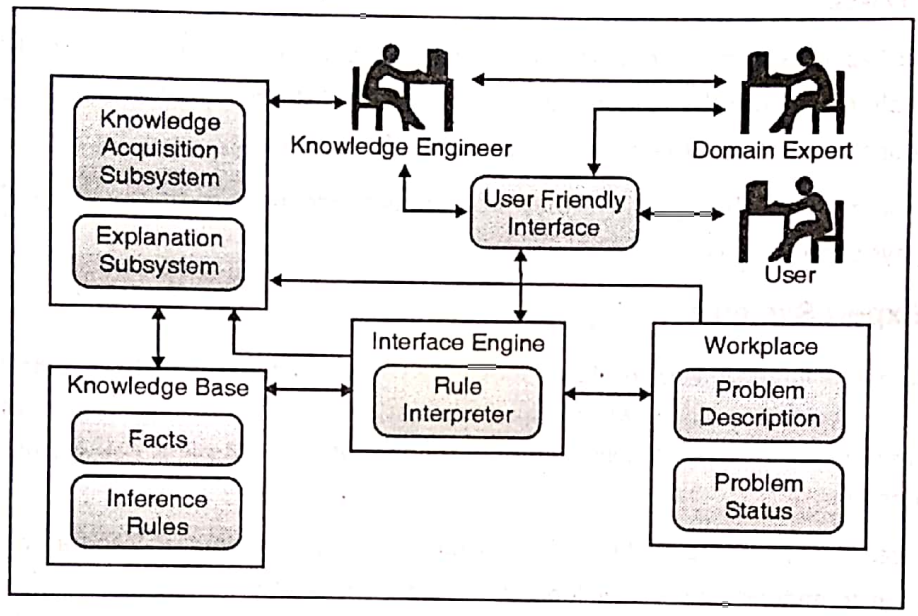

Fig. 6.3.1 : Expert System Architecture

- If we consider inference engine as the brain of the expert systems, then knowledge base is the heart. As the heart is more powerful, the brain can function faster and efficient way. Hence the success of any expert system is more or less depends on the quality of knowledgebase it works on.

1. Knowledge base

There are two types of knowledge expert systems can have about the task domain.

(i) Factual knowledge
(ii) Heuristic knowledge

(i) Factual knowledge : It is the knowledge which is accepted widely as a standard knowledge. It is available in text books, research journals and internet. It is generally accepted and verified by domain experts or researchers of that particular field.

(ii) Heuristic knowledge : It is experiential, judgmental and may not be approved or acknowledged publically. This type of knowledge is rarely discussed and is largely individualistic. It doesn't have standards for evaluation of its correctness. It is the knowledge of good practice, good judgment and probable reasoning in the domain. It is the knowledge that is based on the "art of good guessing." It is very subjective to the practitioner's knowledge, and experience in the respective problem domain.

The knowledge base an expert uses is based on his learning from various sources over a time period. Hence, the knowledge store of an expert increases with number of years of experience in the given field, which allows him to interpret the information in his databases to advantage in diagnosis, analysis and design.

2. Inference engine

The inference engine has a problem solving module which organizes and controls the steps required to solve the problem. A common but powerful inference engine involves chaining of IF...THEN rules to form a line of reasoning. There are two types of chaining practices to solve a problem.

1. **Forward Chaining:** This type of reasoning strategy starts from a set of conditions and moves toward some conclusion, also called as data driven approach.
2. **Backward Chaining :** Backward chaining is a goal driven approach. In this type of reasoning, the conclusion is known and the path to the conclusion needs to be found out. For example, a goal state is given, but the path to that state from start state is not known, then backward reasoning is used.

Inference engine is nothing but these methods implemented as program modules. Inference engine manipulates and uses knowledge in the knowledge base to generate a line of reasoning.

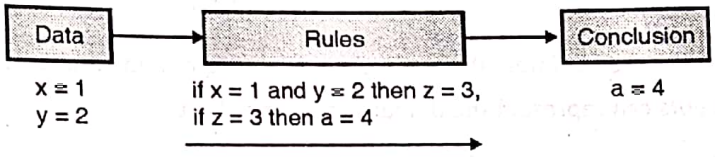

Fig. 6.3.2 : Forward Chaining

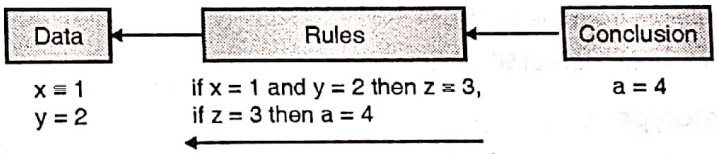

Fig. 6.3.3 : Backward Chaining

Knowledge is most of the times incomplete and uncertain. There are various ways to deal with uncertainty in knowledge. One of the simplest methods is to associate a weight or a confidence factor with each rule. The set of methods used to represent uncertain knowledge in combination with uncertain data in the reasoning process is called reasoning with uncertainty. "Fuzzy Logic" is an important area of artificial intelligence which provides methods for reasoning with uncertainty and the systems that use it are called as "fuzzy systems".

3. **Explanation subsystem :**Most of the times, human experts can guess or use their gut feeling to approximate or estimate the results. As an expert system try to mimic human thinking, it uses uncertain or heuristic knowledge as we humans do. As a result, its credibility of the system is often in question, as is the case with humans. As an answer to a problem itself is questionable, one urges to know the rationale or the reasoning behind the answer. If the rationale seems to be acceptable, generally the answer is considered to be correct. So is the case with expert systems!!

Most of the expert systems have the ability to answer questions of the form: "Why is the answer X?" Inference engine can generate explanations by tracing the line of reasoning used by it.

4. **Knowledge engineer :** Building an expert system is also known as "knowledge engineering" and its practitioners are called knowledge engineers. The primary job of knowledge engineer is to make sure that the expert system has all the knowledge needed to solve a problem. He does a vital task of gathering knowledge from domain experts.

Knowledge engineer needs to learn how the domain expert reasons with their knowledge by interviewing them. Then he translates his knowledge into programs using which he designs the interface engine. There might be some uncertain knowledge involved in the knowledgebase; knowledge engineer needs to decide how to integrate this with the available knowledgebase. Lastly, he needs to decide upon what kind of explanations will be required by the user, and according to that he designs the inference levels.

6.4 Development Phases of Expert Systems

To develop an expert system following are the general steps followed. It is an iterative process. Steps in developing an expert system include:

1. 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.

2. 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.

3. 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.

4. 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.

5. 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.

6. Maintain the ES

- Keep the knowledge base up-to-date by regular review and update.
- Cater for new interfaces with other information systems, as those systems evolve.

6.5 Representing and Using Domain Knowledge

- Knowledge affects the development, efficiency, speed, and maintenance of the system. Knowledge representation is a way to transform human knowledge to machine understandable format. It is a very challenging task in expert systems, as the knowledge is very vast, unformatted and most of the times it is uncertain. Knowledge representation formalizes and organizes the knowledge required to build an expert system.

- The knowledge engineer must identify one or more forms in which the required knowledge should be represented in the system. He must also ensure that the computer can use the knowledge efficiently with the selected reasoning methods. As the quality of knowledge matters, the representation used also matters a lot as that will best allow a programmer to write the code for the system.
- A number of knowledge-representation techniques have been devised till date to represent the knowledge efficiently, but finally it depends on the application, the design of system. Few of the knowledge representation techniques are mentions below.
  - Production rules
  - Decision trees
  - Semantic nets
  - Factor tables
  - Attribute-value pairs
  - Frames
  - Scripts
  - Logic
  - Conceptual graphs
- Out of these the most commonly used methods for representing domain knowledge are Production Rules, Semantic Nets and Frames. Let's have detail study of these methods.

a. Production Rules

- One widely used representation of knowledge is the set of production rules, or simply rule tree. A rule has a condition and an action associated with it. The condition part is identified by keyword "IF". It lists a set of conditions in some logical combination. Actions are specified in "THEN" part.
- As the IF part of the rule is satisfied; consequently, the THEN part actions can be taken. The piece of knowledge represented by the production rule is used to produce the line of reasoning. Expert systems whose knowledge is represented in rule form are called rule-based systems. We have studied rule-based agents named as simple reflex agents in chapter 1.
- As human thinking is evolved on the basis of situation→ conclusion or condition→action basis, this model is predominantly used representing knowledge in ES.
- IF-THEN rules are the simplest and efficient way to represent expert's knowledge. It takes following form.

IF $a_1, a_2, a_3, \ldots, a_n$ THEN $b_1, b_2, b_3, \ldots b_n$

Where $a_1, a_2, \ldots$ are the situation or conditions and

$b_1, b_2, \ldots$ are the conclusions or actions.

Consider the following example :

Design advisor : [Steele et al., 1989] is a system that critiques chip designs. Its rules look like :
If : the sequential level count of ELEMENT is greater than 2,
UNLESS the signal of ELEMENT is resetable
then : critique for poor resetability
DEFEAT : poor resetability of ELEMENT
due to : sequential level count of ELEMENT greater than 2
by : ELEMENT is directly resettable

b. Semantic Nets

- A semantic net or semantic network is a knowledge representation technique used for propositional information, so it is also called a propositional net. In semantic networks the knowledge is represented as objects and relationships between objects. They are two dimensional representations of knowledge. It conveys meaning. Relationships provide the basic structure for organizing knowledge. It uses graphical notations to draw the networks. Mathematically a semantic net can be defined as a labelled directed graph. As nodes are associated with other nodes semantic nets are also referred to as associative nets.
- Semantic nets consist of nodes, links and link labels. Objects are denoted by nodes of the graph while the links indicate relations among the objects. Nodes can appear as circles or ellipses or rectangles to represent objects such as physical objects, concepts or situations. Links are drawn as arrows to express the relationships between objects, and link labels specify specifications of relationships.
- The two nodes connected to each other via a link are related to each other. The relationships can be of two types: "IS-A" relationship or "HAS" relationship. IS-A relationship stands for one object being "part of" the other related object. And "HAS" relationship indicates one object "consists of" the other related object. These relationships are nothing but the super class subclass relationships. It is assumed that all members of a subclass will inherit all the properties of their super classes. That's how semantic network allows efficient representation of inheritance reasoning.

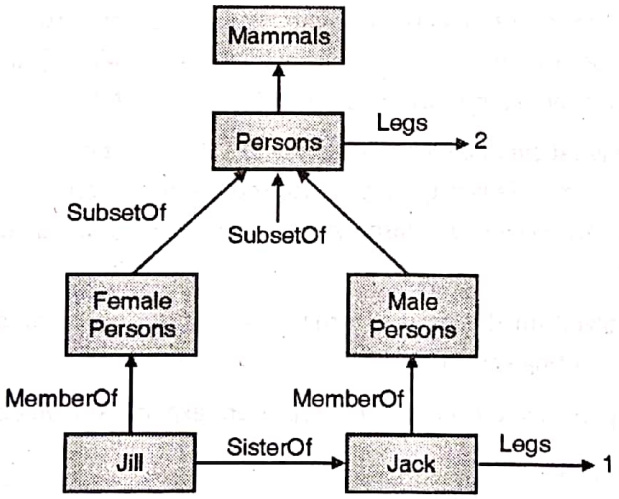

Fig. 6.5.1:Semantic net example

- For example Fig. 6.5.1 is showing an instance of a semantic net. In the Fig. 6.5.1 all the objects are within rectangles and connected using labeled arcs. The links are given labels according to the relationships. This makes the network more readable and conveys more information about all the related objects. For example, the "Member Of" link between Jill and Female Persons indicates that Jill belongs to the category of Female Persons.
- It does also indicate the inheritance among the related objects. Like, Jill inherits the property of having two legs as she belongs to the category of Female Persons which in turn belongs to the category of Persons which has a boxed Legs link with value 2.

- Semantic nets also can represent multiple inheritance through which, an object can belong to more than one objects and an object can be a subset of more than one another objects. It does also allows a common form of inference known as inverse links. The inverse links make the job of inference algorithms much easier to answer reverse queries.
- For example, in Fig. 6.5.1 there IS-A HAS Sister link which is the inverse of Sister Of link. If there is a query "such as who the sister of Jack ?"The inference system will discover that Has Sister is the inverse of Sister of, to make the inference algorithm follow the link HAS Sister from Jack to Jill and answer the query.

Advantages of Semantic Nets

1. Semantic nets are very expressive and minute details can be represented using semantic nets. For example, it can represent default values for categories. In Fig. 6.5.1, Jack has one leg can be represented by explicitly mentioning as shown. This specified value replaces the default value 2 for number of legs for person object.
2. Semantic nets can represent semantics of relationships in a transparent manner.
3. Semantic nets are simple and easy to understand.
4. Semantic nets are easy to implement using PROLOG.

Disadvantage of Semantic Nets

1. The links between the objects represent only binary relations. In relations where more than two objects are involved cannot be represented using semantic nets. For example, the sentence Run(Chennai Express, Chennai, Bangalore, Today) cannot be represented directly.
2. There is no standard definition of link names.

c. Frames

- Frames provide a convenient structure for representing objects that are typical to stereotypical situations. For example, visual scenes, structure of complex physical objects etc. In this technique knowledge is decomposed into highly modular format.
- Frame is a type of schema used in many Artificial Intelligence applications including vision and natural language processing. Frames are also useful for representing commonsense knowledge.
- As frames can represent concepts, situations, attributes of concepts, relationships between concepts, and also procedures to explain their relationships. It allows nodes to have structures and hence is regarded as 3-D representations of knowledge.
- A frame is also known as unit, schema, or list. Typically, a frame consists of a list of properties of the object and associated values for the properties ; similar to the fields and values; also called as slots and slot fillers. The contents of slot can be a string, numbers, functions, procedures, etc.
- A frame is a group of slots and fillers that defines a stereotypical object. Rather than a single frame, frame systems usually have collection of frames connected to each other. One of the attribute values can be another frame. For example, Fig. 6.5.2 shows a frame for a book object.

Slots	Fillers
Publisher	Techmax
Title	Intelligent Systems
Author	PurvaRaut
Edition	First
Year	15
Pages	275

Fig. 6.5.2 : A simple Frame for Book Object

- This is one of the simplest frames, but frames can have more complex structure in real world. A powerful knowledge system can be built with filler slots and inheritance provided by frames. Following Fig. 6.5.3 is the example for generic frame.

Slot	Fillers
Name	Laptop
specialization_of	a_kind_of machine
Types	(desktop, laptop, mainframe, super) if-added : Procedure ADD_LAPTOP
Speed	default: faster if-needed : Procedure FIND_SPEED
Location	(home, office, mobile)
under_warranty	(yes, no)

Fig.6.5.3 : Generic Frame Example

- From Fig.6.5.3 we can conclude that fillers are of various types and it may also include procedural attachments. Let's see what those types are.
  1. **Value :** "laptop" in the "name" slot.
  2. **Range :** "types" slot has (desktop, laptop, mainframe, super) as fillers.
  3. **If-needed :** It's a procedural attachment. It will be executed when a filler value is needed.
  4. **Default :**"Default" value is taken if no other value exists. It represents common sense knowledge, when no specific value is available.
  5. **If-added :** It's a procedural attachment. It is required if any value is to be added to a slot. In the above example, on arrival of a new type of laptop, ADD_LAPTOP procedure should be executed to add that information.
  6. **If-removed :**It's a procedural attachment. It is used to remove a value from the slot.
- Finally, the domain knowledge in an expert system will reside in a Knowledge Representation Language (KRL), such as an expert system shell. Let's understand what it is.

6.6 Expert System-Shell

- In early days each of the expert system was built from scratch. As numbers of expert systems are developed, researchers found that there are few parts of the systems which were common.
- As in each of those systems, there was an interpreter, which was developed separately for each system to interpret the rules. It was vividly noticed that, there can be a common generic interpreter developed independent of the domain.
- Also, there are only a handful of ways to represent knowledge, or to make inferences, or to generate explanations; it is better to have a generic system without any domain specific knowledge.
- Such systems are known as skeletal systems, shells, or simply AI tools. A new ES can be developed by adding domain knowledge to the shell.

– Fig. 6.6.1 depicts generic components of expert system shell. It includes Knowledge acquisition system, Knowledgebase, Inference engine, Explanation subsystem, and user interface. Knowledgebase and inference mechanism are the core components of shell.

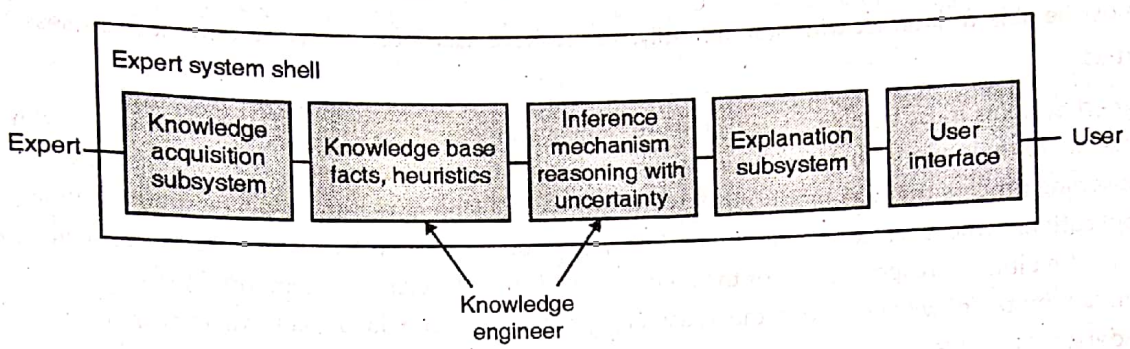

Fig. 6.6.1 : Components of Expert System Shell

– Let us understand in short what each component is, and what it is used for.

1. **Knowledge acquisition system :** Knowledge acquisition is the first and the fundamental step in building an expert system. It helps to collect the experts knowledge required to solve the problems and build the knowledgebase. But this is also the biggest bottleneck of building ES.
2. **Knowledge base :** This component is the heart of expert system. It stores all factual and heuristic knowledge about the application domain. It provides with the various representation techniques for all the data, as the programmers are required to program the system accordingly.
3. **Inference mechanism :** Inference engine is the brain of expert system. This component is mainly responsible for generating inference from the given knowledge from the knowledgebase and produce line of reasoning in turn the result of the user's query.
4. **Explanation subsystem :** This part of shell is responsible for explaining or justifying the final or intermediate result of user query. It is also responsible to justify need of additional knowledge.
5. **User interface :** It is the means of communication with the user. Earlier it was not use to be a part of expert systems, as there was no significance associated with user interface. But later on it was also recognized as an important component of the system, as it decides the utility of expert system.

– Building expert systems by using shells has significant advantages. It is always advisable to use shell to develop expert system as it avoids building the system from scratch. To build an expert system using system shell, one needs to enter all the necessary knowledge about a task domain into a shell. The expert can himself create the knowledgebase by undergoing some training on how to use the shell. The inference engine that applies the knowledge to the given task is built into the shell.

– There are many commercial shells available today, "EMYCIN" derived from MYCIN ; "Inter modeller" used to develop educational expert systems, to name a few. There are variety of sizes for shells, starting from shells on PCs, to shells on large mainframe computers. They range in complexity from simple, forward-chained, rule-based systems. Accordingly to the size and complexity, shell price range from hundreds to tens of thousands of dollars. Application wise, they range from general-purpose shells to customized ones, such as financial planning or real-time process control.

6.7 Explanations

- Expert systems are developed with the aim of efficient and maximum utilization of technology in place of human expertise. To achieve this aim, along with the accuracy in the working, also the user interface must be good.
- User must be able to interact with system easily. To facilitate user interactions, system must possess following two properties:
  1. Expert systems must be able to explain its reasoning. In many cases user are interested in not only knowing the answers to their queries but also how the system has generated that answer. That will ensure the accuracy of the reasoning process that has produced those answers. Such kind of reasoning will be required typically in medical applications, where doctor needs to know why a particular medicine is advised for a particular patient, as he owns the ultimate responsibility for the medicines he subscribe. Hence, it is important that the system must store enough meta-knowledge about the reasoning process and should be able to explain it to the user in an understandable way.
  2. The system should be able update its old knowledge by acquiring new knowledge. As the knowledgebase is where the system's power resides, expert system should be able to maintain the complete, accurate and up to date knowledge about the domain. It is easy said than done!! As the system is programmed based on the available knowledgebase, it is very difficult to adapt to the changes in knowledgebase. It must have some mechanism through which the programs will learn its expert behavior from raw data. Another comparatively simple way is to keep on interacting with human experts and update the system.
- TEIRESIES was the first expert system with both these properties implemented in it. MYCIN used TEIRESIES as its user interface.
- As the TEIRESIAS-MYCIN system answers the user questions, he might be satisfied or might want to know the reasoning behind the answers. User can very well find it out by asking "HOW" question.
- The system will interpret it as "How do you know that?" and answers it by using backward chining starting from the answer to one of the given fact or rule. TEIRESIAS-MYCIN can do fairly good job in satisfying the user's query and providing proper reasoning for it.

Types of Explanations

There are four types of explanations generally the expert system is asked for there are:

1. Report on rule trace on progress of consultation.
2. Explanation on how the system has reached to a particular conclusion.
3. Explanation of why the system is asking question.
4. Explanation of why the system did not give any conclusion.

6.8 Knowledge Acquisition

- Knowledge is the most important ingredient in any expert system. The power of expert systems resides in the specific, high quality knowledge they contain about task domains. The more quality knowledge a system is given, the more competent it becomes.
- Even if expert systems shells simplify programming by providing a general skeleton; knowledge acquisition has to be done separately for each system. The choice of reasoning method, or a shell, is important, but it's equally important to accumulate high quality knowledge for developing any expert system. Knowledge engineers perform the task of knowledge acquisition.

- The knowledge acquisition component allows the expert to enter their knowledge or expertise into the expert system. It can be refined as and when required. Nowadays automated systems that allow the expert to interact directly with the system are becoming increasingly common, thereby reducing the work pressure of knowledge engineer.
- The knowledge acquisition process has three principal steps. Fig. 6.8.1 depicts them in a diagrammatic form. They are as follow :
  1. **Knowledge elicitation :** Knowledge engineer needs to interact with the domain expert and get all the knowledge. He also needs to format it in a systematic way so that it can be used while developing the expert system shell.
  2. **Intermediate knowledge representation :** The knowledge obtained from the domain expert needs to be store in some intermediate representation, such that, it can be worked upon to produce the final refined version.
  3. **Knowledgebase representation :** The intermediate representation of the knowledge needs to be complied and transformed into an executable format. This version of knowledge is ready to get uploaded to system shell as it is. e.g. production rules, that the inference engine can process.

     In the process of expert system development, numbers of iterations through these three stages are required in order to equip the system with good quality knowledge.
- The iterative nature of the knowledge acquisition process can be represented in Fig. 6.8.1.

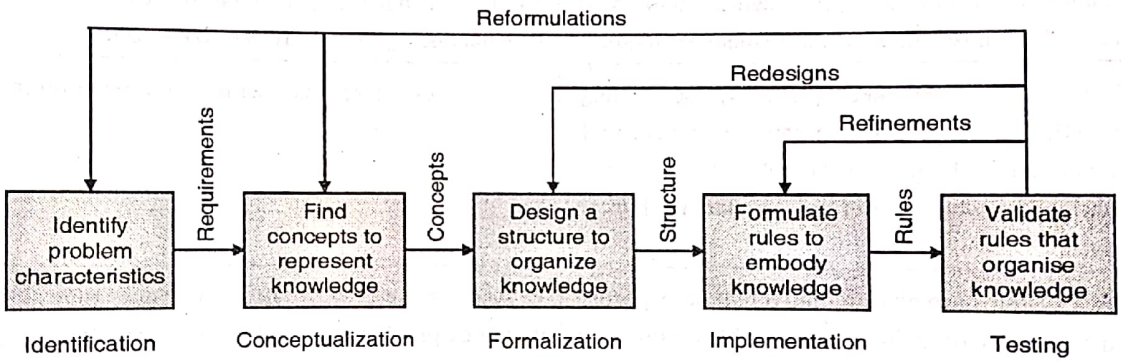

Fig. 6.8.1 : Stages of knowledge acquisition

- As quality of knowledge base determines the success of an expert system, AI researchers are continually exploring and adding new methods of knowledge representation and reasoning. Future of expert systems depends on breaking the knowledge acquisition bottleneck and codifying and representing a large knowledge infrastructure.

6.8.1 Knowledge Elicitation

- The knowledge elicitation is the first step of knowledge acquisition. In this process itself there are several stages. Generally knowledge engineer performs these steps.
- These steps need to be carried out before meeting the domain expert to collect the quality knowledge. They are as follows.
  1. Gather maximum possible data about the problem domain from books, manuals, internet, etc., in order to become familiar with specialist terminology and jargons of the problem domain.
  2. Identify the types of reasoning and problem solving tasks that the system will be required to perform.
  3. Find domain expert or team of experts who are willing to work on the project. Sometimes experts are frightened of being replaced by a computer system!
  4. Interview the domain experts multiple times during the course of building the system. Find out how they solve the problems that, the system is expected to solve. Have them check and refine the intermediate knowledge representation.

– Knowledge elicitation is a time consuming process. There exists automated knowledge elicitation and machine learning techniques which are increasingly being used as common modern alternatives.

6.9 Expert System vs. Traditional System

Sr. No.	Expert System (ES)	Traditional System (TS)
1.	Expert system Is knowledgebase + inference Engine	Traditional systems are Algorithms + data structures
2.	Expert systems can predict future events based on the current data input patterns using their inference process.	TS cannot do prediction tasks so efficiently as they do not have a strong inference engine.
3.	ES have very strong inference system to deduce knowledge from given facts	TS does not have a strong inference system to deduce knowledge.
4.	ES have explanation subsystem which can explain and justify the results at any intermediate stage.	TS do not have any mechanism to justify the results. Manual debugging is required to be done
5.	ES can do tasks like planning, scheduling, prediction, diagnosis; which require to deal with current data input and knowledge from past experiences, which generally handled by human experts.	TS cannot do expert tasks without human intervention.
6.	Expert systems are able to match human expertise in a particular domain provided with a complete knowledgebase and a powerful inference engine.	TS systems can only provide data based on available data, It cannot provide user with knowledge about the domain. Hence human expertise are required to analyse the data further to deduce knowledge from it. Hence TS cannot eliminate Human experts it can only assist them.

6.10 The Applications of Expert Systems

– The applications of expert systems find their way into most areas of knowledge work. Expert systems are used in the place of human interactions to industrial and commercial problems.

– There exists a wide spectrum of applications for expert systems. These applications are so big in number that, they cannot be categorized easily. All these applications are clustered into seven major classes.

1. **Diagnosis and troubleshooting of devices and systems of all kinds**

   This category of applications comprises of expert systems that detect faults and suggest corrective actions for a malfunctioning device or process. Medical diagnosis was one of the first knowledge areas to which expert system technology was applied.

2. **Planning and scheduling**

   These expert systems analyze set of goals and provides with the plan of action in order to achieve the predetermined goal. These applications has great commercial potential. For example, airline flight scheduling, manufacturing job-shop scheduling; and manufacturing process planning.

3. **Configuration of manufactured objects from subassemblies**

Configuration is historically one of the most important of expert system applications. In this category of applications a solution to a problem is synthesized from a given set of elements related by a set of constraints. These applications are used in many different industries. Examples include modular home building, manufacturing, complex engineering design and manufacturing.

4. **Financial decision making**

Expert systems are used vigorously in financial services, typically in foreign exchange trading. Advisory programs are widely used to assist bankers to take decisions about loans. Insurance companies used expert systems to assess the risk associated with the customer in order to determine price for the insurance. A typical application in the financial markets is

5. **Knowledge publishing**

Usage of expert systems in this area is relatively new, but has good potential for further exploration. There are many applications based on user information access preferences.

6. **Process monitoring and control**

In this category systems performing analysis of real time data are designed. These systems obtain data from physical devices and produce results specifying anomalies, predicting trends, etc. We have existing expert systems to monitor the manufacturing processes in the steel making and oil refining industries.

7. **Design and manufacturing**

These expert systems assist in the design of physical devices and processes. It is ranging from high level conceptual design of abstract entities all the way to factory floor configuration of manufacturing processes.

Review Questions

Q. 1 Explain the need of hybrid approach.

Q. 2 Explain Fuzzy neural system.

Q. 3 Explain the architecture of ANFIS with the help of block diagram.

Q. 4 What is expert system? Give examples of real time expert systems.

Q. 5 What are the characteristics of expert system?

Q. 6 Explain in building blocks of expert system.

Q. 7 What are the various methods to represent domain knowledge in case of expert system?

Q. 8 Write a short note on expert system shell.

Q. 9 How reasoning is performed in case of expert systems?

Q. 10 What is knowledge acquisition? What are the steps of knowledge acquisition?

Q. 11 Write a short note on knowledge elicitation.

❑❑❑