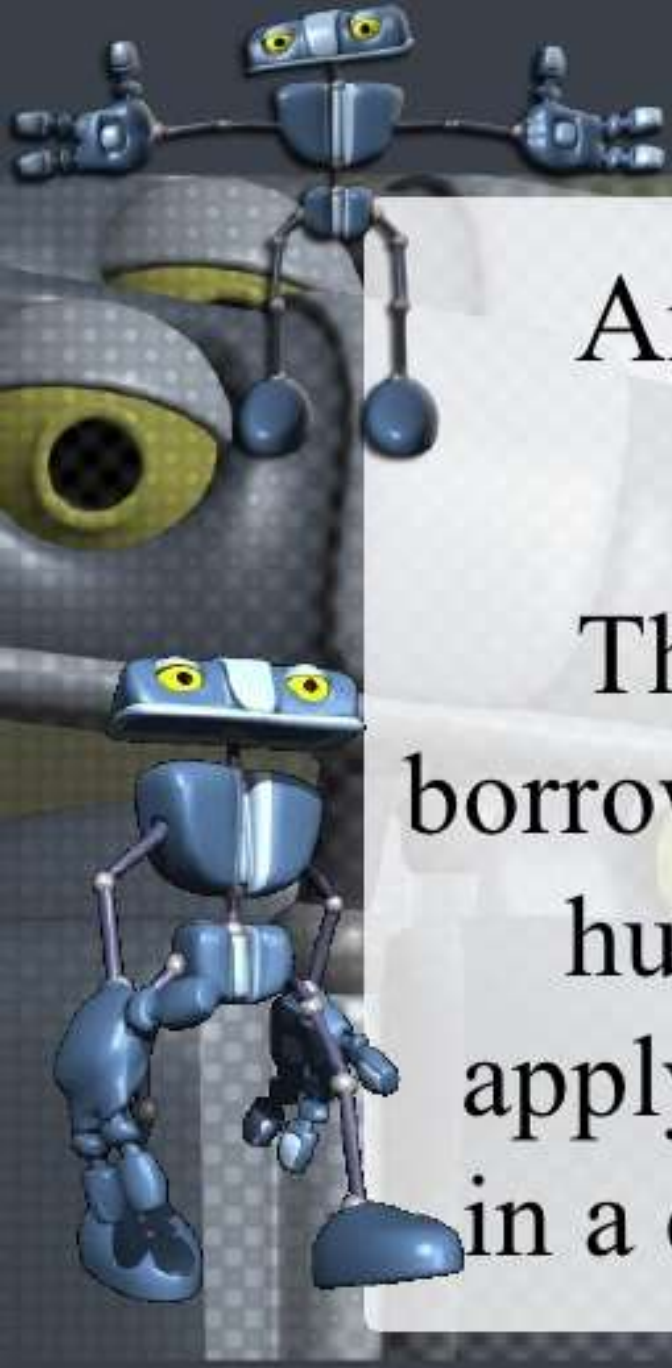
Artificial Intelligence

BCA Final Yr



What is A.I. ?

Artificial Intelligence

This generally involves borrowing characteristics from human intelligence, and applying them as algorithms in a computer friendly way.

Artificial Intelligence

- Intelligence: "ability to learn, understand and think" (Oxford dictionary)
- AI is the study of how to make computers make things which at the moment people do better.
- Examples: Speech recognition, Smell, Face, Object, Intuition, Inferencing, Learning new skills, Decision making, Abstract thinking

IMPORTANCE OF AI

Why Artificial Intelligence?

- Computers are well suited to perform mechanical computations.
- Unlike humans, computers have trouble understanding specific situations, and adapting to new situations.
- Artificial Intelligence aims to improve machine behavior in tackling complex tasks.

IMPORTANCE AND ADVANTAGES OF AI

- AI can replace humans being in some more specific jobs for some business store and household day to day activities and will help to sort out the manpower
- AI machines can help in hospitals, providing food and medicines where human being feared to be attacked by such disease. Robotics is good example
- It is estimated that AI will help human being in aeronautics to know the universe.

Future Of Artificial Intelligence

- Future of AI is really unknown
- Most jobs will be done by robots within 30 years, says professor Moshe Vardi of rice university , Texas(USA)
- It can make humans extinct or immortal
- The optimists hope that we'll one day invent a superintelligence that solves every problem we can imagine.

CONCEPTS
Technologies / techniques used

- Machine learning
- Waypoint Graph
- Continuous Control
- Cellular Automata
- Case Based Reasoning
- Blend Database
- Dynamic Programming
- Finite State Machines
- Fuzzy Logic
- Influence Mapping
- Hierarchical Task Network
- A* Algorithms
- Ragdoll
- Teleo Reactive Programs
- Scripting
- Motion Graph
- Motion Capture
- Navigation Mesh
- Navigation Graph
- Artificial Neural Networks
- Policy Search
- Path Planning
- Proportional Derivative Controller
- Path Planning
- Path Finding
- Path Following
- Steering
- Semantic Markup
- Reinforcement Learning
- Alpha-Beta pruning
- MiniMax

Types of Artificial Intelligence System

- The techniques & software that enables computers to mimic human behavior in various ways.
- 4 types of AI;
  - Expert System – reasons thru problems & advice in a form of conclusion/ recommendations
  - Neural Networks – can be programmed to recognize patterns
  - Genetic Algorithms – imitates evolution characteristics, implement the process of suitability to produce better decision.
  - Intelligent Agents – move around computer/networks to perform repetitive process on its own & adapt to your preferences.

Criteria for success

- long term: Turing Test (for Weak AI)
  - as proposed by Alan Turing (1950), if a computer can make people think it is human (i.e., intelligent) via an unrestricted conversation, then it is intelligent
  - Turing predicted fully intelligent machines by 2000, *not even close*
  - Loebner Prize competition, extremely controversial

- short term: more modest success in limited domains
  - performance equal or better than humans
    e.g., game playing (Deep Blue), expert systems (MYCIN)
  - real-world practicality $$$
    e.g., expert systems (XCON, Prospector), fuzzy logic (cruise control)

Criteria of success

- Tasks define clearly
- Test the possible solutions
- Solution is analysis
- Implement procedure of define task

Uses in different fields:-

As AI is a very vast field it is used in every field

- ✓Computer
- ✓Banking and faineance
- ✓Robotics
- ✓Transport
- ✓Medical
- ✓Education
- ✓Games
- ✓Music
- ✓Speech Recognition

To build a system to solve a problem

1. Define the problem precisely
2. Analyse the problem
3. Isolate and represent the task knowledge that is necessary to solve the problem
4. Choose the best problem-solving techniques and apply it to the particular problem.

Formal Description of the problem

1. Define a state space that contains all the possible configurations of the relevant objects.
2. Specify one or more states within that space that describe possible situations from which the problem solving process may start (initial state)
3. Specify one or more states that would be acceptable as solutions to the problem. (goal states)
4. Specify a set of rules that describe the actions (operations) available.

Production Systems

A production system consists of:

- A set of rules, each consisting of a left side that determines the applicability of the rule and a right side that describes the operation to be performed if that rule is applied.

- One or more knowledge/databases that contain whatever information is appropriate for the particular task. Some parts of the database may be permanent, while other parts of it may pertain only to the solution of the current problem.

- A control strategy that specifies the order in which the rules will be compared to the database and a way of resolving the conflicts that arise when several rules match at once.

- A rule appli[illegible]

Production system

Inorder to solve a problem:

- We must first reduce it to one for which a precise statement can be given. This can be done by defining the problem's state space (start and goal states) and a set of operators for moving that space.
- The problem can then be solved by searching for a path through the space from an initial state to a goal state.
- The process of solving the problem can usefully be modelled as a production system.

Control Strategies

- How to decide which rule to apply next during the process of searching for a solution to a problem?
- The two requirements of good control strategy are that
  - it should cause motion.
  - It should be systematic

Breadth First Search

- Algorithm:
  1. Create a variable called NODE-LIST and set it to initial state
  2. Until a goal state is found or NODE-LIST is empty do
     a. Remove the first element from NODE-LIST and call it E. If NODE-LIST was empty, quit
     b. For each way that each rule can match the state described in E do:
        i. Apply the rule to generate a new state
        ii. If the new state is a goal state, quit and return this state
        iii. Otherwise, add the new state to the end of NODE-LIST

Algorithm: Depth First Search

1. If the initial state is a goal state, quit and return success
2. Otherwise, do the following until success or failure is signaled:
   a. Generate a successor, E, of initial state. If there are no more successors, signal failure.
   b. Call Depth-First Search, with E as the initial state
   c. If success is returned, signal success. Otherwise continue in this loop.

Heuristic Search

- A Heuristic is a technique that improves the efficiency of a search process, possibly by sacrificing claims of completeness.
- Heuristics are like tour guides
- They are good to the extent that they point in generally interesting directions;
- They are bad to the extent that they may miss points of interest to particular individuals.
- On the average they improve the quality of the paths that are explored.
- Using Heuristics, we can hope to get good (though possibly nonoptimal) solutions to hard problems such asa TSP in non exponential time.
- There are good general purpose heuristics that are useful in a wide variety of problem domains.
- Special purpose heuristics exploit domain specific knowledge

Example Simple Heuristic functions

- Chess : The material advantage of our side over opponent.
- TSP: the sum of distances so far
- Tic-Tac-Toe: 1 for each row in which we could win and in we already have one piece plus 2 for each such row in we have two pieces

Problem Characteristics

- Inorder to choose the most appropriate method for a particular problem, it is necessary to analyze the problem along several key dimensions:
  - Is the problem decomposable into a set of independent smaller or easier subproblems?
  - Can solution steps be ignored or at least undone if they prove unwise?
  - Is the problem's universe predictable?
  - Is a good solution to the problem obvious without comparison to all other possible solutions?
  - Is the desired solution a state of the world or a path to a state?
  - Is a large amount of knowledge absolutely required to solve the problem or is knowledge important only to constrain the search?
  - Can a computer that is simply given the problem return the solution or will the solution of the problem require interaction between the computer and a person?

Generate-and-Test

- It is a depth first search procedure since complete solutions must be generated before they can be tested.
- In its most systematic form, it is simply an exhaustive search of the problem space.
- Operate by generating solutions randomly.
- Also called as British Museum algorithm
- If a sufficient number of monkeys were placed in front of a set of typewriters, and left alone long enough, then they would eventually produce all the works of shakespeare.
- Dendral: which infers the struture of organic compounds using NMR spectrogram. It uses plan-generate-test.

Generate-and-Test

- Algorithm:
  1. Generate a possible solution. For some problems, this means generating a particular point in the problem space. For others it means generating a path from a start state
  2. Test to see if this is actually a solution by comparing the chosen point or the endpoint of the chosen path to the set of acceptable goal states.
  3. If a solution has been found, quit, Otherwise return to step 1.

Issues in the design of search programs

- The direction in which to conduct the search (forward versus backward reasoning).
- How to select applicable rules (Matching)
- How to represent each node of the search process (knowledge representation problem)

Simple Hill Climbing

Algorithm

1. Evaluate the initial state.
2. Loop until a solution is found or there are no new operators left to be applied:
   - Select and apply a new operator
   - Evaluate the new state:
     goal → quit
     better than current state → new current state

Hill Climbing

- Is a variant of generate-and test in which feedback from the test procedure is used to help the generator decide which direction to move in search space.
- The test function is augmented with a heuristic function that provides an estimate of how close a given state is to the goal state.
- Computation of heuristic function can be done with negligible amount of computation.
- Hill climbing is often used when a good heuristic function is available for evaluating states but when no other useful knowledge is available.

Hill Climbing

- To implementation of heuristic search is through a procedure called ***hill climbing***
- Hill climbing strategies expand the current state in the search and evaluate its children
- the best child is selected for further expansion; neither its siblings nor its parent are retained
- search halts when it reaches a state that is better than any of its children
- go uphill along the steepest possible path until it can go no farther
- because it keeps no history, the algorithm cannot recover from failures of its strategy

Best First Search

- Combines the advantages of bith DFS and BFS into a single method.
- DFS is good because it allows a solution to be found without all competing branches having to be expanded.
- BFS is good because it does not get branches on dead end paths.
- One way of combining the tow is to follow a single path at a time, but switch paths whenever some competing path looks more promising than the current one does.

Best First Search

- Heuristic based search technique.
- Every node in the search space has an Evaluation function (heuristic function)associated with it.
- Evaluation function==heuristic cost function (in case of minimization problem) OR objective function(in case of maximization).
- Decision of which node to be expanded depends on value of evaluation function.
- Evaluation value= cost/distance of current node from goal node.
- For goal node evaluation function value=0

Problem characterization

- A **constraint satisfaction problem** (or CSP) is a special kind of problem that satisfies some additional structural properties beyond the basic requirements for problems in general.
- In a CSP, the states are defined by the values of a set of **variables** and the goal test specifies a set of **constraints** that the values have to obey.

Unit-2

Knowledge Representation

- Knowledge representation and reasoning is the fields of artificial intelligence (AI) dedicated to representing information about the world in a form that computer system can utilize to solve complex tasks such as diagonasing a medical condition or having a dialogue in natural language.Read more on Brainly.in - https://brainly.in/question/5721712#readmore

Knowledge Representation

- **What to Represent?**

Let us first consider what kinds of knowledge might need to be represented in AI systems:

- **Objects**
  - -- Facts about objects in our world domain. *e.g.* Guitars have strings, trumpets are brass instruments.
- **Events**
  - -- Actions that occur in our world. *e.g.* Steve Vai played the guitar in Frank Zappa's Band.

- **Performance**
  - -- A behavior like *playing the guitar* involves knowledge about how to do things.
- **Meta-knowledge**
  - -- knowledge about what we know.

Thus in solving problems in AI we must represent knowledge and there are two entities to deal with:

- **Facts**
  - -- truths about the real world and what we represent. This can be regarded as the *knowledge level*
- **Representation of the facts**
  - which we manipulate. This can be regarded as the *symbol level* since we usually define the representation in terms of symbols that can be manipulated by programs.

Approaches to Knowledge Representation

Simple relational knowledge

- The simplest way of storing facts is to use a relational method where each fact about a set of objects is set out systematically in columns. This representation gives little opportunity for inference, but it can be used as the knowledge basis for inference engines.
- Simple way to store facts.
- Each fact about a set of objects is set out systematically in columns.
- Little opportunity for inference.
- Knowledge basis for inference engines.

Figure: Simple Relational Knowledge

Musician	Style	Instrument	Age
Miles Davis	Jazz	Trumpet	deceased
John Zorn	Avant Garde	Saxophone	35
Frank Zappa	Rock	Guitar	deceased
John Mclaughlin	Jazz	Guitar	47

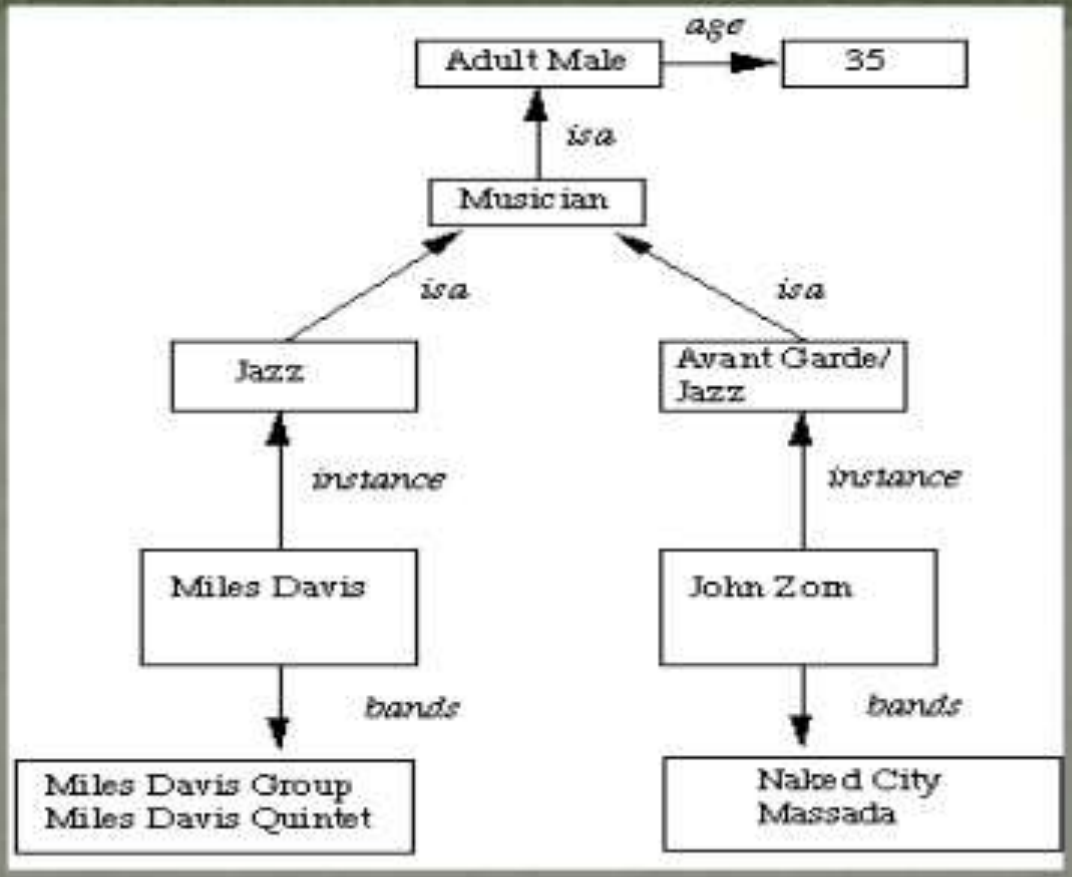
Inheritable knowledge

Relational knowledge is made up of objects consisting of

- attributes
- corresponding associated values.

We extend the base more by allowing inference mechanisms:

- **Property inheritance**
  - **elements inherit values from being members of a class.**
  - **data must be organized into a hierarchy of classes.**



Boxed nodes -- objects and values of attributes of objects.
Values can be objects with attributes and so on.
Arrows -- point from object to its value.
This structure is known as a **slot and filler structure, semantic network or a collection of frames.**

Inferential Knowledge

Represent knowledge as *formal logic*:

- *All dogs have tails*

V(X) : $dog(x) \rightarrow hasatail(x)$

Advantages:

- A set of strict rules.
  - Can be used to derive more facts.
  - Truths of new statements can be verified.
  - Guaranteed correctness.
- Popular in AI systems. *e.g* Automated theorem proving

Procedural Knowledge

Basic idea:

- Knowledge encoded in some procedures
  - small programs that know how to do specific things, how to proceed.

- **What are the issues in knowledge representation?**
  The fundamental goal of Knowledge Representation is to facilitate inferencing (conclusions) from knowledge. The issues that arise while using KR techniques are many. Some of these are explained below.

- · Important Attributes : Any attribute of objects so basic that they occur in almost every problem domain ?

- · Relationship among attributes: Any important relationship that exists among object attributes ?

- · Choosing Granularity : At what level of detail should the knowledge be represented ?

- · Set of objects : How sets of objects be represented ?

- · Finding Right structure : Given a large amount of knowledge stored, how can relevant parts be accessed ?

PREDICATE LOGIC

- Can represent objects and quantification
- Theorem proving is semi-decidable

Representing simple facts (Preposition)

"SOCRATES IS A MAN"
SOCRATESMAN ---------1

"PLATO IS A MAN"
PLATOMAN ---------2

Fails to capture relationship between Socrates and man.
We do not get any information about the objects involved
Ex:

if asked a question : "who is a man?" we cannot get answer.

Using Predicate Logic however we can represent above facts as: Man(Socretes) and Man(Plato)

Representing Simple Facts in Logic

To answer the question

Was Marcus loyal to Caesar?

To produce a formal proof , reasoning backward from th e desired goal

- loyalto(Marcus, Caesar)

To prove the goal, rules of inference are to be used to trans form into another goal that in turn be transformed and s o on, until there are no insatisfied goals remaining.

Representing simple facts in logic

- It is raining.
  RAINING
- If it is raining then it is not sunny.
  RAINING → ~ SUNNY
- All men are motal.
  MORTALMAN
  man(mortal)
  $\forall X : man(X) \rightarrow mortal(X)$

1. Change sentences 1-8 on page 134 to predicate logic.
2. prove ~loyalto(Marcus,Caesar) on page 136 Fig. 5.2

Instance and Isa relationship

- " Marcus is a man"

  man(marcus)

  OR

  instance(marcus , man) where marcus is an object/ instance of class 'man'

" all pompeians were romans"

$\forall x$: pompeian(x) $\rightarrow$ roman(x).

OR

$\forall x$: instance(x, pompeian) $\rightarrow$ instance(x, roman).

- **Isa Predicate :**

" all pompeians were romans"

$$\forall x: \text{pompeian}(x) \rightarrow \text{roman}(x).$$

OR

$$\forall x: \text{instance}(x, \text{pompeian}) \rightarrow \text{instance}(x, \text{roman}).\text{------}(1)$$

- Now using isa predicate (1) becomes,

$$\text{Isa}(\text{pompeian}, \text{roman})$$

which means pompeian is a subclass of roman class

but it also requires extra axiom :

$$\forall x: \forall y: \forall z: \text{isa}(y, z) \wedge \text{instance}(x, y) \rightarrow \text{instance}(x, z)$$

Computable functions and predicates

- " Marcus was born in 40 A.D"

  Born(Marcus, 40)

- " All Pompeians died when volcano erupted in 79 A.D"

  Erupted(volcano, 79) ∧ ∀x: [Pompeian (x) → Died (x , 79)]

- **" no mortal lives longer than 150 years"**
- **How to solve ?**
- **let t1 is time instance 1 and t2 is time instance 2**
- **We use computable function gt(… , ….) which computes greater than.**

∀x: ∀t1: ∀t2: mortal (x) ∧ born (x, t1) ∧ gt(t2 –t1, 150) → dead (x, t2)

Computable Functions and Predicates

•All the simple facts can be expressed as combination of indivi dual predicates such as

tryassassinate(Marcus, Caesar)

•This is fine if the number of facts is not very large. But suppo se if we want to express simple facts such as greater-than and less-than relationships:

Q Computable predicates

greater-than(1, 0)	less-than(0, 1)
greater-than(2, 1)	less-than(1, 2)
greater-than(3, 2)	less-than(2, 3)
....	....

Unit-3

Syntactic Analysis

- Syntax concerns the proper ordering of words and its affect on meaning
- This involves analysis of the words in a sentence to depict the grammatical structure of the sentence
- The words are transformed into structure that shows how the words are related to each other
- Eg. "the girl the go to the school". This would definitely be rejected by the English syntactic analyzer

Syntactic Analysis

- Syntax mapped into semantics
  - Nouns ↔ things, objects, abstractions.
  - Verbs ↔ situations, events, activities.
  - Adjectives ↔ properties of things, ...
  - Adverbs ↔ properties of situations, ...
- A parser recovers the phrase structure of an utterance, given a grammar (rules of syntax)
- Parser's outcome is the structure (groups of words and respective parts of speech)

Syntactic Analysis

- Rules of syntax (grammar) specify the possible organization of words in sentences and allows us to determine sentence's structure(s)
  - John saw Mary with a telescope
    - John saw (Mary with a telescope)
    - John (saw Mary with a telescope)
- Parsing: given a sentence and a grammar
  - Checks that the sentence is correct according with the grammar and if so returns a parse tree representing the structure of the sentence

What is natural language processing?

- Process information contained in natural language text
- Also known as Computational Linguistics (CL), Human Language Technology (HLT), Natural Language Engineering (NLE)

NLP for machines...

- Analyze, understand and generate human languages just like humans do
- Applying computational techniques to language domain
- To explain linguistic theories, to use the theories to build systems that can be of social use
- Started off as a branch of Artificial Intelligence
- Borrows from Linguistics, Psycholinguistics, Cognitive Science & Statistics
- Make computers learn our language rather than we learn theirs

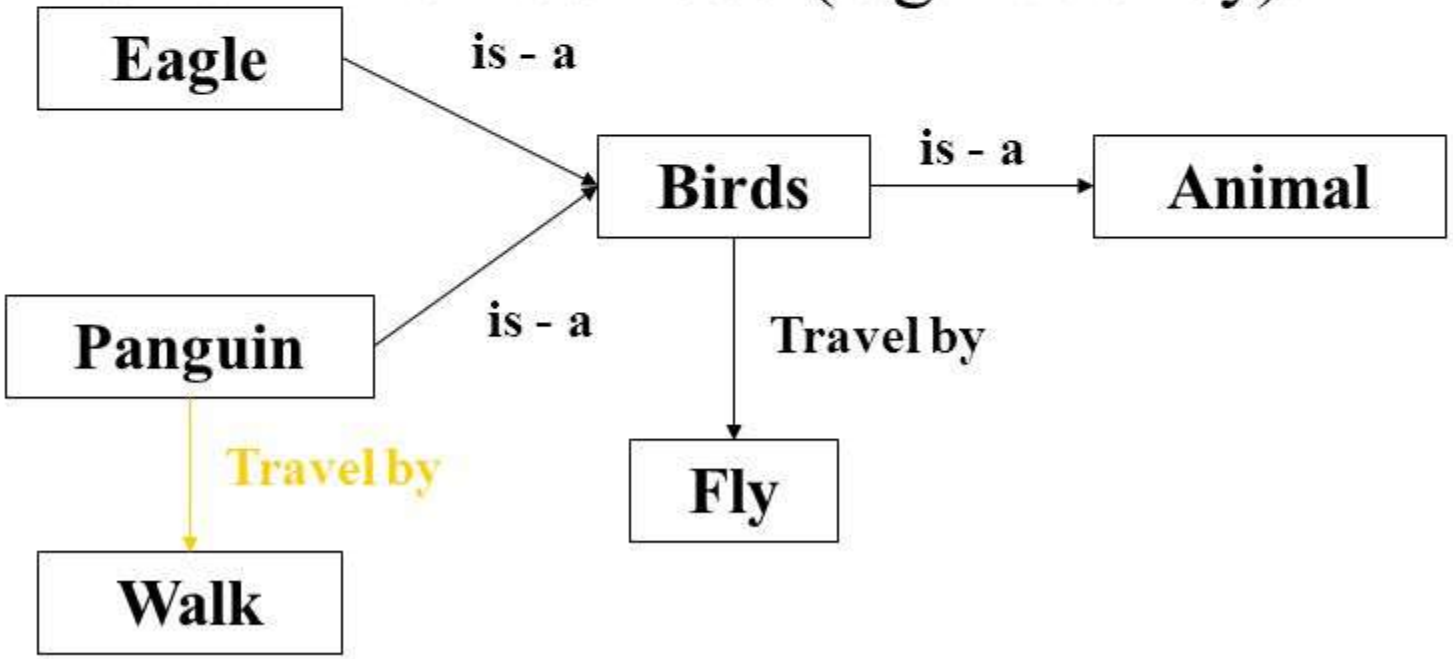
Semantic Analysis

- Semantics concerns the (literal) meaning of words, phrases, and sentences
- This abstracts the dictionary meaning or the exact meaning from context
- The structures which are created by the syntactic analyzer are assigned meaning
- E.g.. "colorless blue idea" .This would be rejected by the analyzer as colorless blue do not make any sense together

Semantic Networks

Relations

- Some times we had to override a relation for an inherited node (e.g travel by).



Discourse Analysis

. The meaning of an individual sentence may depends on the sentence that precede it and may influence the meaning of sentences that follow it.

e.g "john wanted it" the word 'it' depends upon john.

Pragmatic analysis

. It derives knowledge from external commonsense information.

. It means understanding purposeful use of language in situation.

e.g " DO you know what time it is?"
should be interpreted as a request.

Discourse and pragmatic processing

- Parts of actions. Consider the text:
  - John went on a business trip to New Yrok.
  - He left on an early morning flight.
  - Taking a flight should be recognized as part of going on a trip.
- Entities involved in actions. Consider the text:
  - My house was broken into last week.
  - They took the TV and the stereo.
  - The pronoun “they” should be recognized as referring to the burglars who broke into the house.
- Elements of sets. Consider the text:
  - The decals we have in stock are stars, the moon, item and a flag.
  - I’ll take two moons.
  - Moons means moon decals

Discourse and Pragmatic processing

- There are a number of important relationships that may hold between phrases and parts of their discourse contexts, including:
- Identical entities. Consider the text:
  - Bill had a red balloon.
  - John wanted it.
  - The word “it” should be identified as referring to red balloon. This type of references are called anaphora.
- Parts of entities. Consider the text:
  - Sue opened the book she just bought.
  - The title page was torn.
  - The phrase “title page” should be recognized as part of the book that was just bought.

What is Learning?

- Learning denotes “changes in the system that are adaptive in the sense that they enable the system to do the same task or tasks drawn from the same population *more efficiently and more effectively* the next time”.

  - Herbert A. Simon, 83

- Agents can *improve their performance* through diligent study of their own experiences.

  - Russell & Norvig

Rote Learning

- Rote learning is a learning technique which focuses on **memorization**. The major practice involved in rote learning is learning by **repetition** by which students commit information to memory in a highly structured way.

Rote Learning:

- In people: straight memorization
- In computer systems: Knowledge engineering; direct entry of rules and facts
- This is all human input. This is the traditional approach to developing ontologies, for instance
- Knowledge base is captured knowledge
- Performer is an inference engine, ontology browser, or other user of the KB
- Learner is the editor use to develop the KB + the human
- Critic is entirely offline, as the human examines or tests the system.

Back to Rote Learning: Some advantages

- Rote methods are routinely used when quick memorization is required, such as learning one's lines in a play or memorizing a telephone number. Rote learning is widely used in the mastery of foundational knowledge.
- Examples of school topics where rote learning is frequently used include phonics in reading, the periodic table in chemistry, multiplication tables in mathematics, anatomy in medicine, cases or statutes in law, basic formulae in any science, etc.

Methods of Learning

- Rote learning:
  - simple storage of computed information
- Learning by taking advice:
  - similar to rote learning,
    but advice may need to be operationalized
- Learning from problem-solving experience:
  - remembering effective structures and methods
- Learning from examples:
  - usually involves a teacher who helps to classify things

Learning by Taking Advice

There are two basic approaches to advice taking:

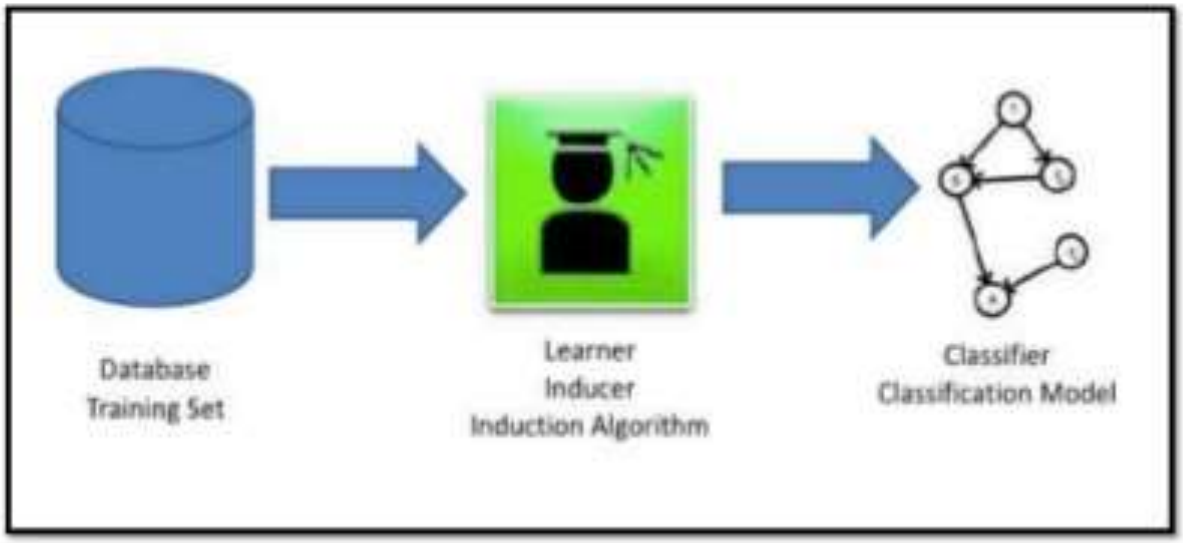
- **Take high level, abstract advice** and convert it into rules that can guide performance elements of the system.
- ***Develop sophisticated tools* such as knowledge base editors and debugging.** These are used to aid an expert to translate his expertise into detailed rules.

Inductive Learning

- A new field of machine learning known as ***inductive learning*** has been introduced to help in inducing general rules and predicting future activities.[4]
- Inductive learning is learning from observation and earlier knowledge by generalization of rules and conclusions. Inductive learning allows for the identification of training data or earlier knowledge patterns.[5]
- The identified and extracted generalized rules come to use in reasoning and problem solving.[6]

Inductive learning

- Given examples of a function (X, F(X))
- Predict function F(X) for new examples X



Application of inductive learning

- The technology for building knowledge-based systems by inductive inference from examples has been demonstrated successfully in several practical applications .[10]
- Inductive learning algorithms are domain independent and can be used in any task involving classification or pattern recognition.[11]

- Making Credit Decisions
- Education
- Medical applications

Why inductive learning

Alternative method of knowledge acquisition in which knowledge learned or induced from examples

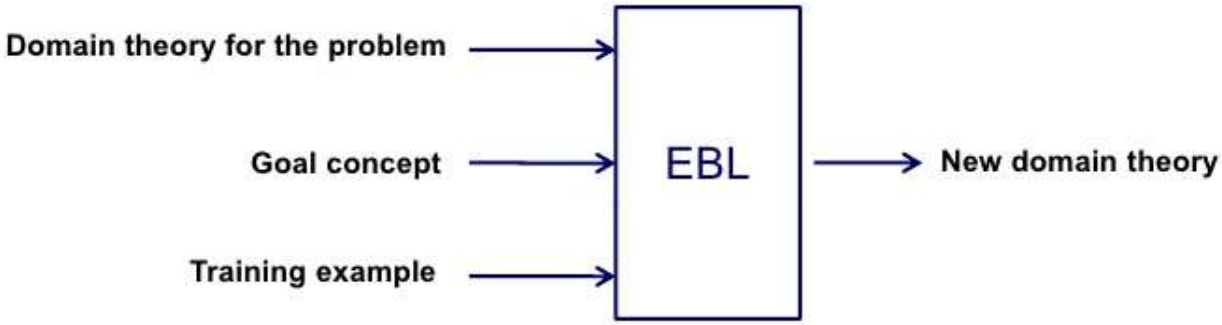
- Human experts are capable of using their knowledge in their daily work, but they usually cannot summaries and generalize their knowledge explicitly in a form which is sufficiently systematic, correct and complete for machine representation and application .[7]
- While it is very difficult for an expert to articulate his knowledge, it is relatively easy to document case studies of the expert's skills at work.[7]

Explanation-based Learning

- Extract general rules from examples
- Basic idea
  - Given an example, construct a proof for the goal predicate that applies using the background knowledge.
  - In parallel, construct a generalized proof with variabilized goal.
  - Construct a new rule, LHS with the leaves of the proof tree and RHS with the variabilized goal.
  - Drop any conditions that are always true regardless of value of variables in the goal.

Explanation-Based Learning

- **Deduce information from a set of observations**
  - Humans learn a lot from few examples
  - Machine: use results from one example to solve the next problem



EXPLANATION-BASED LEARNING (EBL)

Hypothesis^Description |= Classifications
Background |= Hypothesis

- A method to extract general rules from individual observations
- The goal is to *solve a similar problem faster next time.*
- Memoization - speed up by saving results and avoiding solving a problem from scratch
- EBL does it one step further - from observations to rules

Unit-4

What is an expert system?

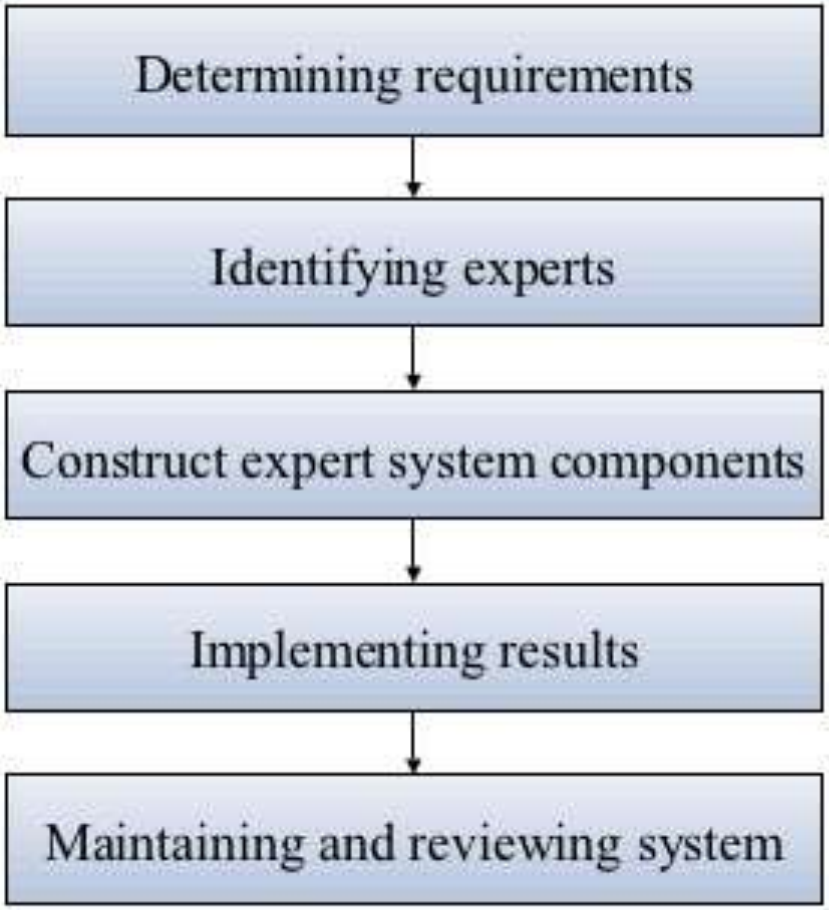
"An expert system is a computer system that emulates, or acts in all respects, with the decision-making capabilities of a human expert."

- Expert Systems = knowledge-based systems = knowledge-based expert systems

EXPERT SYSTEMS

- Expert systems are designed to solve real problems in a particular domain that normally would require a human expert. It can solve many types of problems
- Developing an expert system involves extracting relevant knowledge from human experts in the area of problem, called domain experts.

Expert Systems Development

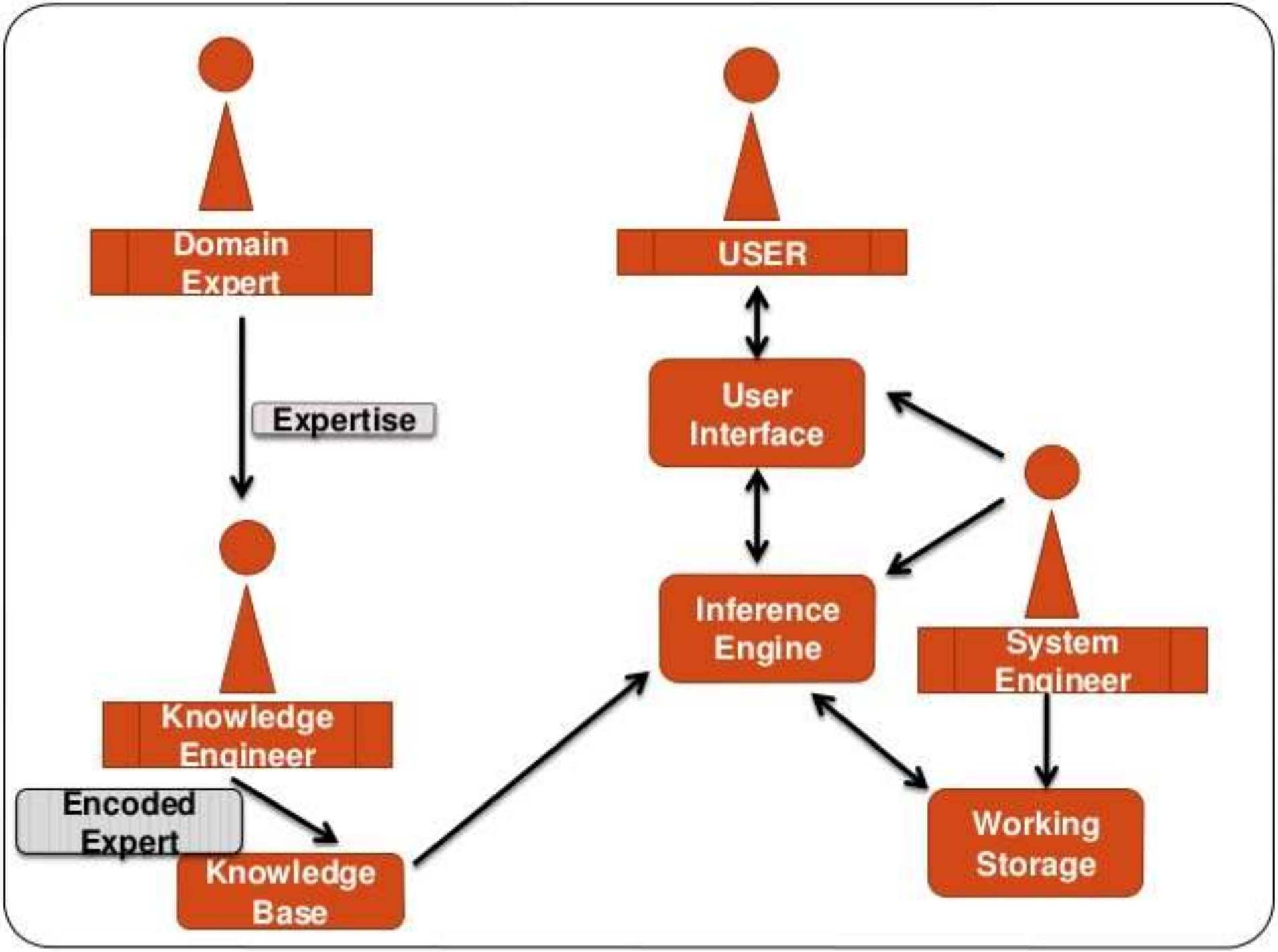


Domain

- The area of knowledge addressed by the expert system.

Applications of Expert Systems and Artificial Intelligence

- Credit granting
- Information management and retrieval
- AI and expert systems embedded in products
- Plant layout
- Hospitals and medical facilities
- Help desks and assistance
- Employee performance evaluation
- Loan analysis
- Virus detection
- Repair and maintenance
- Shipping
- Marketing
- Warehouse optimization


Domain Expert
Expertise
Knowledge Engineer
Encoded Expert
Knowledge Base
USER
User Interface
Inference Engine
System Engineer
Working Storage

Expert Systems

- An expert system is software that attempts to reproduce the performance of one or more human experts, most commonly in a specific problem domain. Ex,
  - Play chess
  - Help in financial decisions
  - Configuration of computers etc

Terms	Importance
Domain expert	The development of accounting software requires knowledge in two different domains, namely accounting and software
Knowledge engineer	KE is an engineering discipline that involves integrating knowledge into computer systems in order to solve complex problems normally requiring a high level of human expertise.
Systems engineering	It is an interdisciplinary field of engineering that focuses on how to design and manage complex engineering projects over their life cycles. Systems engineering deals with work-processes, optimization methods, and risk management tools in such projects.
User	User will be consulting with the system to get advice which would have been provided by the expert

Characteristics Of Expert Systems

- The Highest level of expertise
- Right on time reaction
- Accepting the incorrect reasoning
- Good reliability
- Easily understood
- Flexible
- Symbolic reasoning
- Heuristic reasoning
- Making mistakes
- Expanding with tolerable difficulties

Advantages v/s Disadvantages

Advantages

- Consistent answers for repetitive decisions, processes and tasks
- Holds and maintains significant levels of information
- Encourages organizations to clarify the logic of their decision-making
- Never "forgets" to ask a question, as a human might

Disadvantages

- Lacks common sense
- Cannot make creative responses as human expert
- Domain experts not always able to explain their logic and reasoning
- Errors may occur in the knowledge base
- Cannot adapt to changing environments
- Costly to develop
- Legal & ethical dilemma
- Difficult to use

3. Knowledge Base (Representing and Using Domain Knowledge)

- Transferring knowledge from the human expert to a computer is often the most difficult part of building an expert system.
- The knowledge acquired from the human expert must be encoded in such a way that it remains a faithful representation of what the expert knows, and it can be manipulated by a computer.

- Three common methods of knowledge representation evolved over the years are:
  - IF-THEN rules
  - Semantic networks
  - Frames

Knowledge Representation

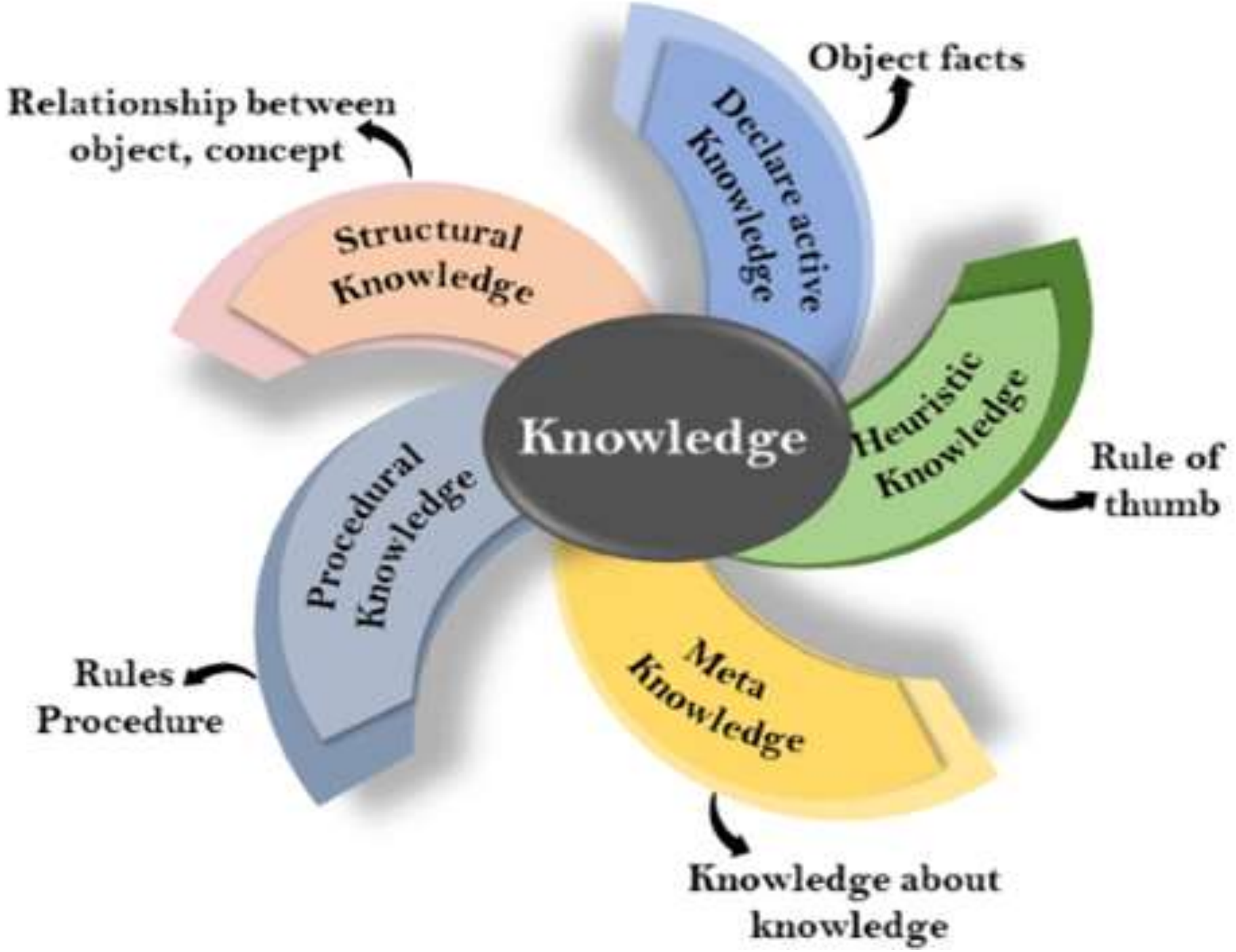
- **What to Represent?**

Let us first consider what kinds of knowledge might need to be represented in AI systems:

- **Objects**
  - -- Facts about objects in our world domain. *e.g.* Guitars have strings, trumpets are brass instruments.
- **Events**
  - -- Actions that occur in our world. *e.g.* Steve Vai played the guitar in Frank Zappa's Band.

Knowledge representation

- AI is the area of computer science focusing on creating machine that can engage on behaviours that humans consider intelligent.
- Knowledge can be managed by knowledge engineering which includes knowledge acquisition, knowledge representation and knowledge manipulation.


Knowledge
Declare active Knowledge
Object facts
Structural Knowledge
Relationship between object, concept
Heuristic Knowledge
Rule of thumb
Procedural Knowledge
Rules Procedure
Meta Knowledge
Knowledge about knowledge

Knowledge Representation

- *Knowledge representation (KR)* is an important issue in both cognitive science and artificial intelligence.
  - In cognitive science, it is concerned with the way people store and process information and
  - In artificial intelligence (AI), main focus is to store knowledge so that programs can process it and achieve human intelligence.
- There are different ways of representing knowledge e.g.
  - predicate logic,
  - semantic networks,
  - extended semantic net,
  - frames,
  - conceptual dependency etc.
- In predicate logic, knowledge is represented in the form of rules and facts as is done in Prolog.

6. Expert System Shells

- Many expert systems are built with products called expert system shells.
- A shell is a piece of software which contains the user interface, a format for declarative knowledge in the knowledge base, and an inference engine.
- The knowledge and system engineers uses these shells in making expert systems.
- ‡ **Knowledge engineer** : uses the shell to build a knowledge base for a particular problem domain.
- ‡ **System engineer** : builds the user interface, designs the declarative format of the knowledge base, and implements the inference engine.

Expert System Shells

- Many expert system s are built with products called expert system shells. A shell is a piece of software which contains the user interface, a format for declarative knowledge in the knowledge base, and an inference engine. The knowledge and system engineers uses these shells in making expert systems.

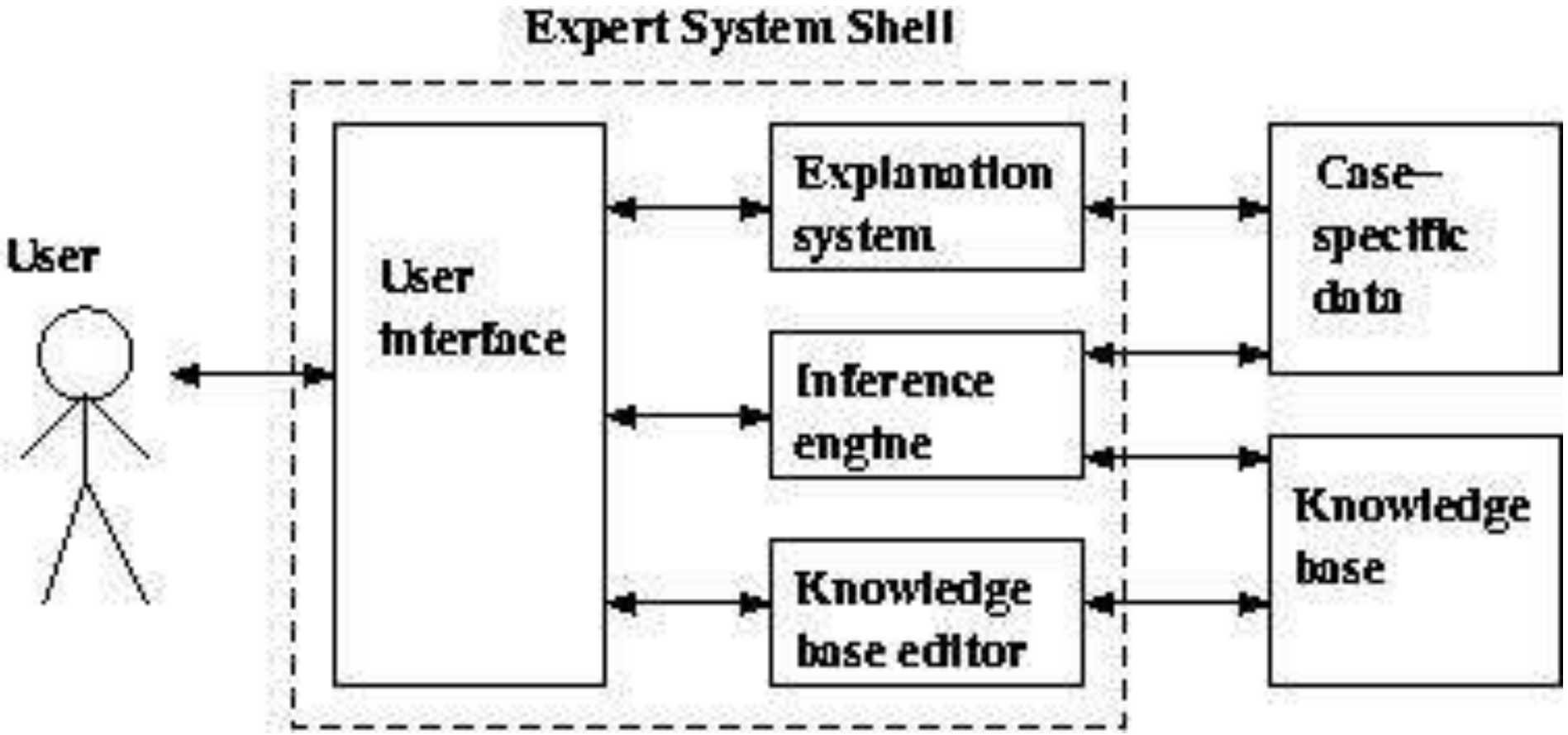
What is an expert system?

"An expert system is a computer system that emulates, or acts in all respects, with the decision-making capabilities of a human expert."

- Expert Systems = knowledge-based systems = knowledge-based expert systems

Advantages of Expert Systems Shells

- **Easy to develop and modify**
- **The use of satisficing**
- **The use of heuristics**
- **Development by knowledge engineers and users**


Expert System Shell
User
User interface
Explanation system
Inference engine
Knowledge base editor
Case-specific data
Knowledge base