

Expert systems

UNIT 4

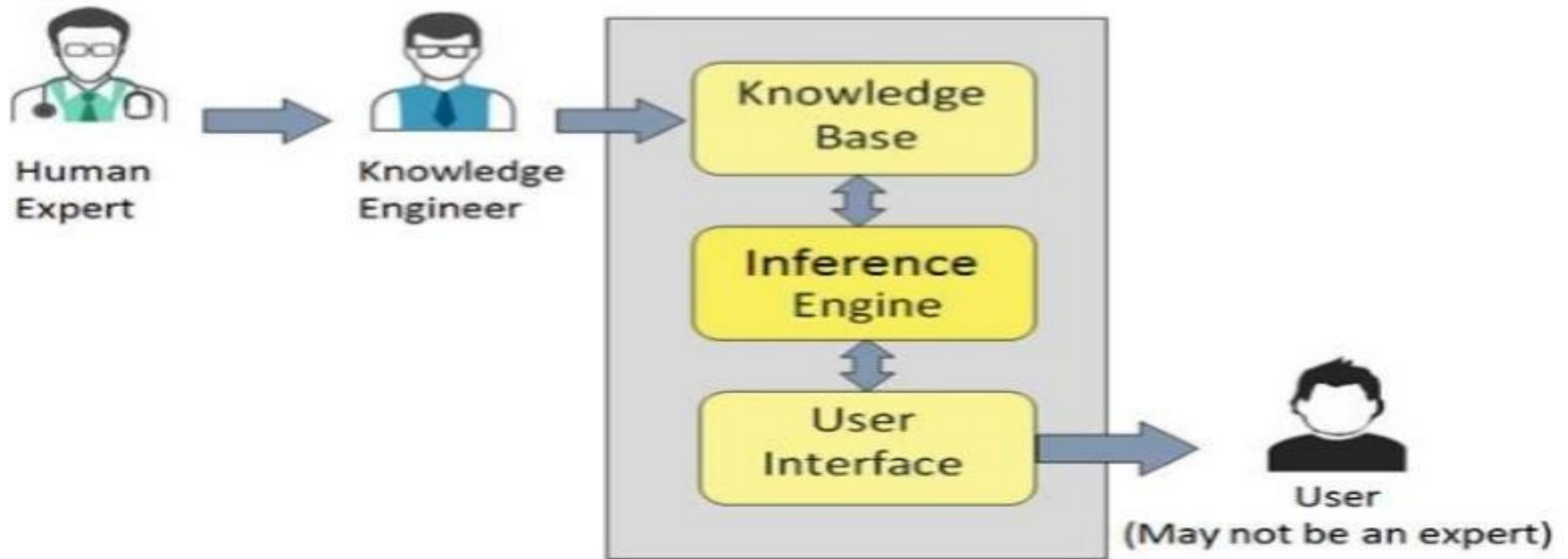
Expert system

- The expert systems are the computer applications developed to solve complex problems in a particular domain, at the level of extra-ordinary human intelligence and expertise.

Expert systems

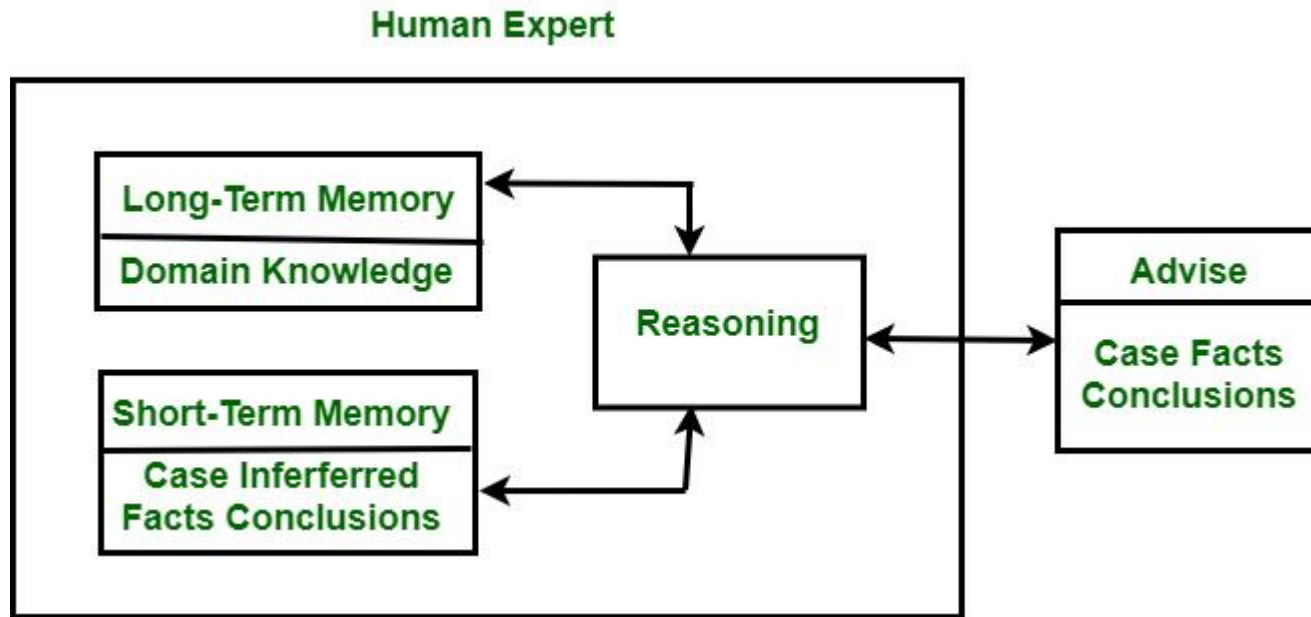
- The other name of expert systems is **knowledge based systems**. They are used for the real world problems like expert quality advice, diagnosis and recommendations.
- Basically, it is a type of computer program that is used to simulate the judgement and behavior of humans or an organization that has an expert knowledge and experience about the particular field.
- Building of an expert system requires a human expert that extract the required knowledge.

Expert System Architecture



Human Expert

- Human expert is an individual who has capability of recognizing the things in a superior way.
- For example: a doctor etc.



Knowledge Base

- It contains domain-specific and high-quality knowledge.
- Knowledge is required to exhibit intelligence. The success of any ES majorly depends upon the collection of highly accurate and precise knowledge.
- What is Knowledge?
- The data is collection of facts. The information is organized as data and facts about the task domain. **Data, information, and past experience** combined together are termed as knowledge.
- **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.

Knowledge representation

- It is the method used to organize and formalize the knowledge in the knowledge base. It is in the form of IF-THEN-ELSE rules.
- **Knowledge Acquisition**
- The success of any expert system majorly depends on the quality, completeness, and accuracy of the information stored in the knowledge base.
- The knowledge base is formed by readings from various experts, scholars, and the **Knowledge Engineers**. The knowledge engineer is a person with the qualities of empathy, quick learning, and case analyzing skills.
- He acquires information from subject expert by recording, interviewing, and observing him at work, etc. He then categorizes and organizes the information in a meaningful way, in the form of IF-THEN-ELSE rules, to be used by interference machine. The knowledge engineer also monitors the development of the ES.

Inference Engine

- Use of efficient procedures and rules by the Inference Engine is essential in deducting a correct, flawless solution.
- In case of knowledge-based ES, the Inference Engine acquires and manipulates the knowledge from the knowledge base to arrive at a particular solution.
- In case of rule based ES, it –
 - Applies rules repeatedly to the facts, which are obtained from earlier rule application.
 - Adds new knowledge into the knowledge base if required.
 - Resolves rules conflict when multiple rules are applicable to a particular case.
 - To recommend a solution, the Inference Engine uses the following strategies –
 - Forward Chaining
 - Backward Chaining

Inference Engine

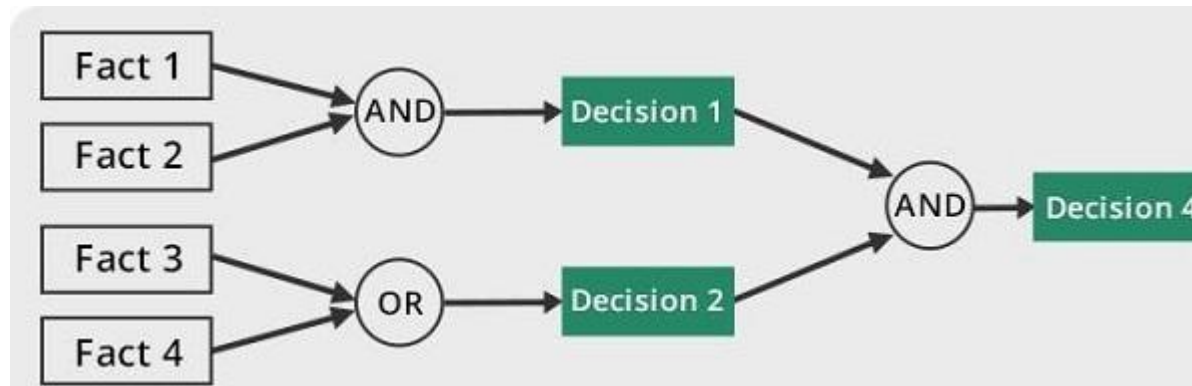
- Use of efficient procedures and rules by the Inference Engine is essential in deducting a correct, flawless solution.
- In case of knowledge-based ES, the Inference Engine acquires and manipulates the knowledge from the knowledge base to arrive at a particular solution.
- In case of rule based ES, it –
 - Applies rules repeatedly to the facts, which are obtained from earlier rule application.
 - Adds new knowledge into the knowledge base if required.
 - Resolves rules conflict when multiple rules are applicable to a particular case.
- To recommend a solution, the Inference 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?**”

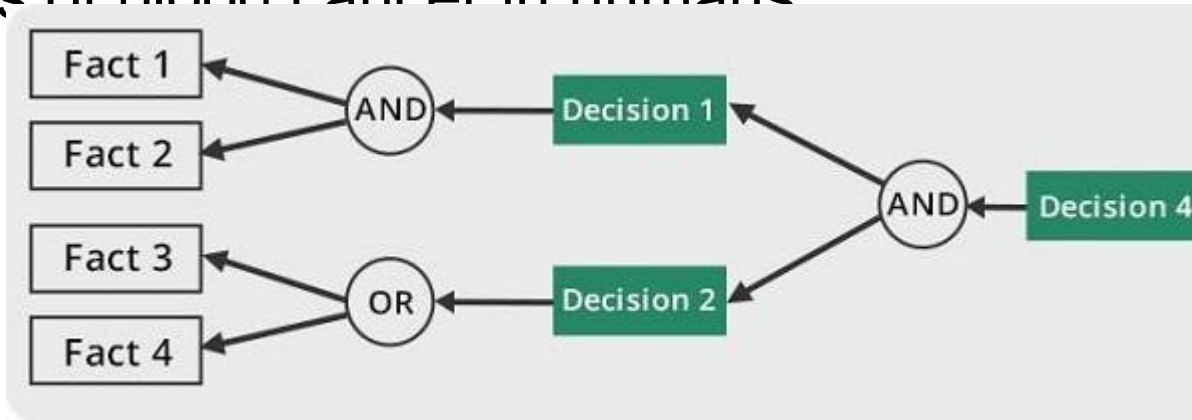
Here, the Inference Engine follows the chain of conditions and derivations and finally deduces the outcome. It considers all the facts and rules, and sorts them before concluding to a solution.

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 Inference 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



User Interface

- User interface provides interaction between user of the ES and the ES itself. It is generally Natural Language Processing so as to be used by the user who is well-versed in the task domain. The user of the ES need not be necessarily an expert in Artificial Intelligence.
- It explains how the ES has arrived at a particular recommendation. 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.
 - The user interface makes it easy to trace the credibility of the deductions.

User Interface

- User interface provides interaction between user of the ES and the ES itself. It is generally Natural Language Processing so as to be used by the user who is well-versed in the task domain. The user of the ES need not be necessarily an expert in Artificial Intelligence.
- It explains how the ES has arrived at a particular recommendation. 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.
 - The user interface makes it easy to trace the credibility of the deductions.

Applications of Expert System

Application	Description
Design Domain	Camera lens design, automobile design.
Medical Domain	Diagnosis Systems to deduce cause of disease from observed data, conduction medical operations on humans.
Monitoring Systems	Comparing data continuously with observed system or with prescribed behavior such as leakage monitoring in long petroleum pipeline.
Process Control Systems	Controlling a physical process based on monitoring.
Knowledge Domain	Finding out faults in vehicles, computers.
Finance/Commerce	Detection of possible fraud, suspicious transactions, stock market trading, Airline scheduling, cargo scheduling.

Participant in Expert Systems Development

Participant	Role
Domain Expert	He is a person or group whose expertise and knowledge is taken to develop an expert system.
Knowledge Engineer	Knowledge engineer is a technical person who integrates knowledge into computer systems.
End User	It is a person or group of people who are using the expert system to get to get advice which will not be provided by the expert.

The process of Building An Expert Systems

- Determining the characteristics of the problem
- Knowledge engineer and domain expert work in coherence to define the problem
- The knowledge engineer translates the knowledge into a computer-understandable language. He designs an inference engine, a reasoning structure, which can use knowledge when needed.
- Knowledge Expert also determines how to integrate the use of uncertain knowledge in the reasoning process and what type of explanation would be useful.

Conventional System vs. Expert System

Conventional System	Expert System
Knowledge and processing are combined in one unit.	Knowledge database and the processing mechanism are two separate components.
The programme does not make errors (Unless error in programming).	The Expert System may make a mistake.
The system is operational only when fully developed.	The expert system is optimized on an ongoing basis and can be launched with a small number of rules.
Step by step execution according to fixed algorithms is required.	Execution is done logically & heuristically.
It needs full information.	It can be functional with sufficient or insufficient information.

Applications of Expert System

Application	Description
Design Domain	Camera lens design, automobile design.
Medical Domain	Diagnosis Systems to deduce cause of disease from observed data, conduction medical operations on humans.
Monitoring Systems	Comparing data continuously with observed system or with prescribed behavior such as leakage monitoring in long petroleum pipeline.
Process Control Systems	Controlling a physical process based on monitoring.
Knowledge Domain	Finding out faults in vehicles, computers.
Finance/Commerce	Detection of possible fraud, suspicious transactions, stock market trading, Airline scheduling, cargo scheduling.

Expert System Technology

- There are several levels of ES technologies available. Expert systems technologies include –
- **Expert System Development Environment** – The ES development environment includes hardware and tools. They are –
 - Workstations, minicomputers, mainframes.
 - High level Symbolic Programming Languages such as **LIS**t **P**rogramming (LISP) and **PRO**grammation en **LOG**ique (PROLOG).
 - Large databases.
- **Tools** – They reduce the effort and cost involved in developing an expert system to large extent.
 - Powerful editors and debugging tools with multi-windows.
 - They provide rapid prototyping
 - Have Inbuilt definitions of model, knowledge representation, and inference design.
- **Shells** – A shell is nothing but an expert system without knowledge base. A shell provides the developers with knowledge acquisition, inference engine, user interface, and explanation facility. For example, few shells are given below –
 - Java Expert System Shell (JESS) that provides fully developed Java API for creating an expert system.

Applications of Expert Systems

- Some popular Application of Expert System:
- Information management
- Hospitals and medical facilities
- Help desks management
- Employee performance evaluation
- Loan analysis
- Virus detection

- Useful for repair and maintenance projects
- Warehouse optimization
- Planning and scheduling
- The configuration of manufactured objects
- Financial decision making Knowledge publishing
- Process monitoring and control
- Supervise the operation of the plant and controller
- Stock market trading
- Airline scheduling & cargo schedules

Examples of AI Expert Systems

1. MYCIN

- MYCIN is amongst the oldest expert systems. It was designed upon the fundamental of backward chaining and was capable to identify infection-causing bacteria.
- MYCIN treats certain bacterial infections and controls acne, additionally to other acne treatments. It prevents infections in people with a history of rheumatic disease, congenital heart condition or other acquired valvular heart condition and who are allergic to penicillin antibiotics.

2. DENDRAL

- An expert system designed to determine the structure of the chemical using its spectrographic data. Its primary aim was to review hypothesis formation and discovery in science.
- The software program DENDRAL is said to be the primary expert system because it automated the decision-making process and problem-solving behavior of organic chemists.

3. R1/XCON

- An Expert System that had the ability to select the best-suited software to perform a particular task assigned by the user.
- A system that ensured the customer was furnished with all the components and software that was needed to form up the required computing system that that they had ordered.

4. PXDES

- Pneumoconiosis X-Ray Diagnosis Expert System (PXDES) is an expert system which is used to diagnose in which stage a patient of lung cancer. The shadow is employed to work out the sort and degree of carcinoma

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.

Semantic Network

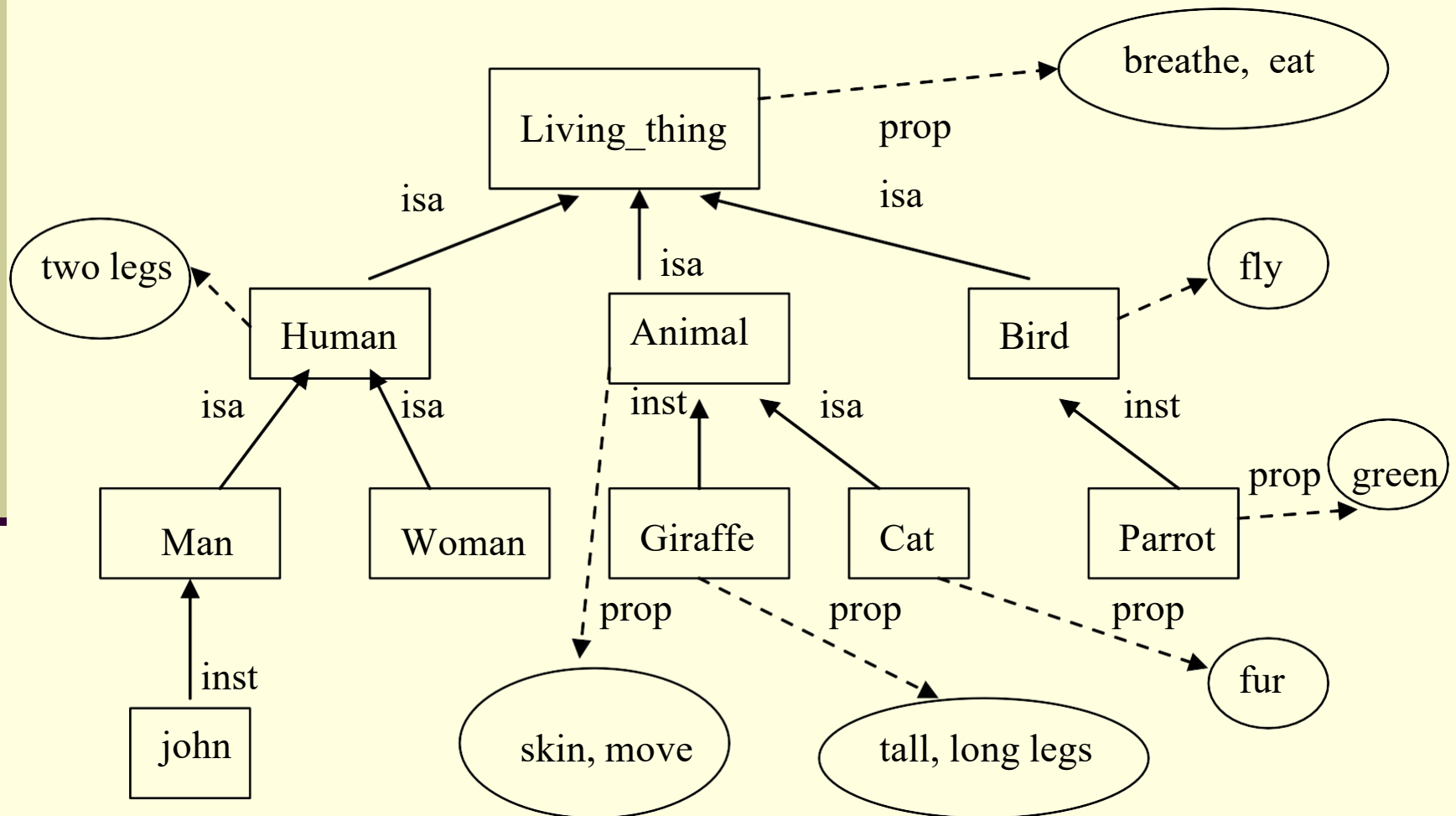
- Formalism for representing information about objects, people, concepts and specific relationship between them.
- The syntax of semantic net is simple. It is a network of labeled nodes and links.
 - It's a directed graph with nodes corresponding to concepts, facts, objects etc. and
 - arcs showing relation or association between two concepts.
- The commonly used links in semantic net are of the following types.
 - **isa** → subclass of entity (e.g., child hospital is subclass of hospital)
 - **inst** → particular instance of a class (e.g., India is an instance of country)
 - **prop** → property link (e.g., property of dog is 'bark')

Representation of Knowledge in Sem Net

“Every human, animal and bird is living thing who breathe and eat. All birds can fly. All man and woman are humans who have two legs. Cat is an animal and has a fur. All animals have skin and can move. Giraffe is an animal who is tall and has long legs. Parrot is a bird and is green in color”.

Representation in Semantic Net

Semantic Net



Property Inheritance Algorithm

Input: Object, and property to be found from Semantic Net;

Output: Yes, if the object has the desired property else return false;

Procedure:

- Find an object in the semantic net; Found = false;
- While {(object \neq root) OR Found } DO
 - { If there is a a property attribute attached with an object then
 - { Found = true; Report 'Yes'}
 - else
 - object=inst(object, class) OR isa(object, class)
- };
- If Found = False then report 'No'; Stop

Queries

- Is john human?
- Is parrot a living thing?
- Is giraffe an animal?
- Is woman subclass of living thing
- Does parrot fly?
- Does john breathe?
- Does parrot have fur?
- Does cat fly?

```
?- instance(john, humans). Y
?- instance (parrot,
    living_thing).          Y
?- instance (giraffe, animal).Y
?- subclass(woman,
    living_things).         Y
?- property(fly, parrot).   Y
?- property (john, breathe). Y
?- property(fur, parrot).   N
?- property(fly, cat).      N
```

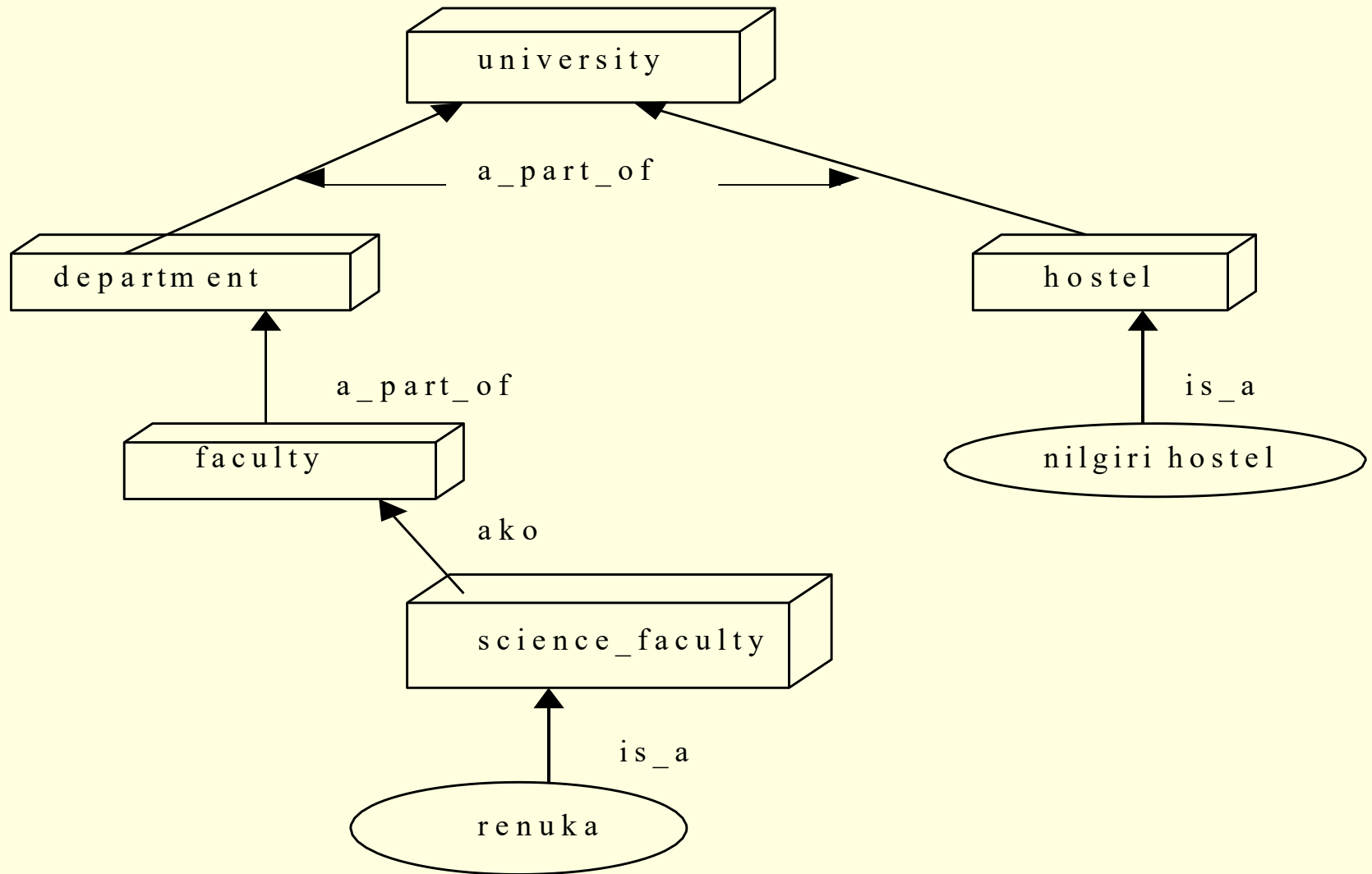
Knowledge Representation using Frames

- Frames are more structured form of packaging knowledge,
 - used for representing objects, concepts etc.
- Frames are organized into hierarchies or network of frames.
- Lower level frames can inherit information from upper level frames in network.
- Nodes are connected using links viz.,
 - **ako / subc** (links two class frames, one of which is subclass of other e.g., science_faculty class is **ako** of faculty class),
 - **is_a / inst** (connects a particular instance of a class frame e.g., Renuka **is_a** science_faculty)
 - **a_part_of** (connects two class frames one of which is contained in other e.g., faculty class **is_part_of** department class).
 - Property link of semantic net is replaced by SLOT fields.

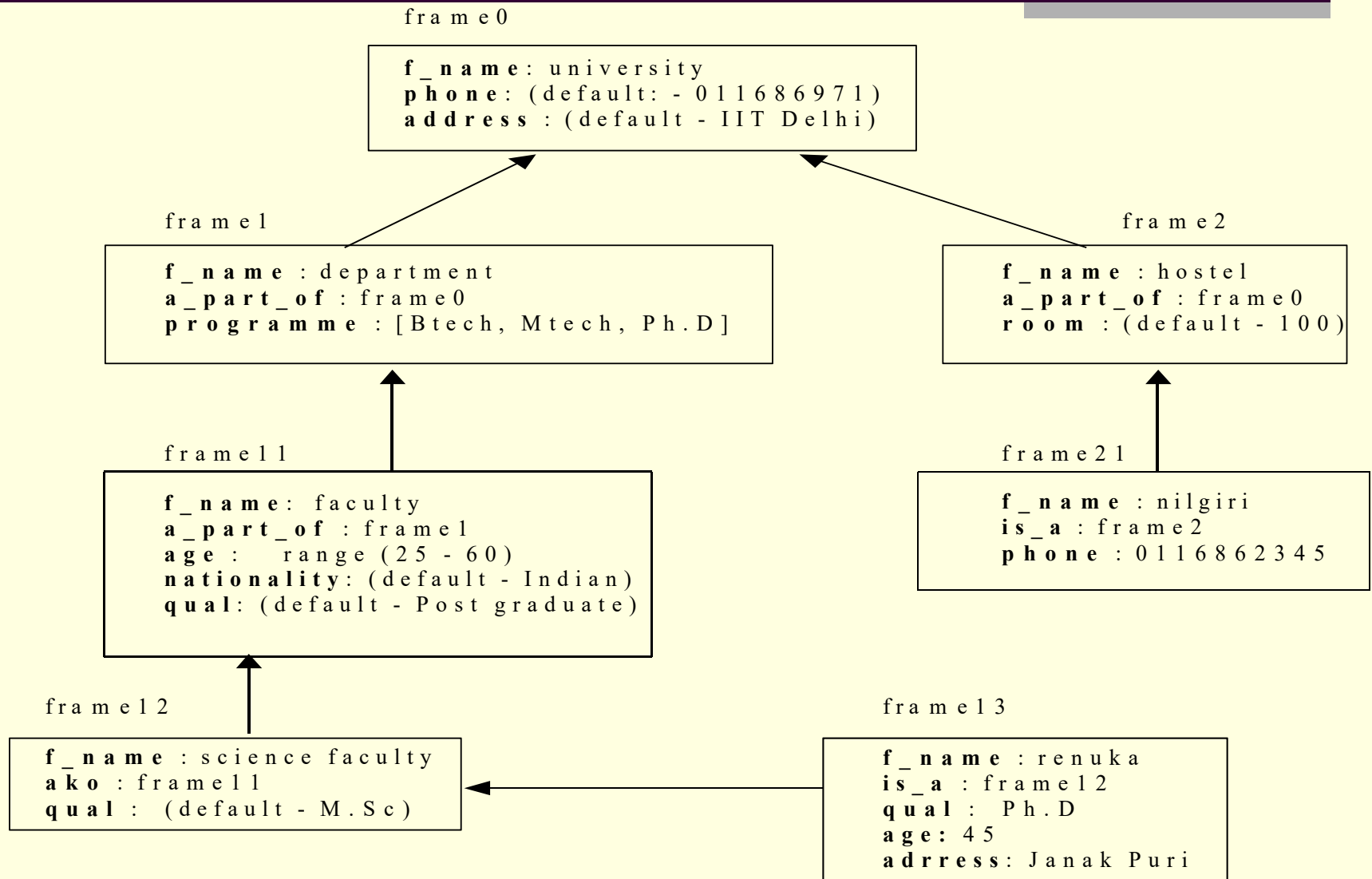
Cont...

- A frame may have any number of slots needed for describing object. e.g.,
 - faculty frame may have name, age, address, qualification etc as slot names.
- Each frame includes two basic elements : slots and facets.
 - Each slot may contain one or more **facets** (called fillers) which may take many forms such as:
 - **value** (value of the slot),
 - **default** (default value of the slot),
 - **range** (indicates the range of integer or enumerated values, a slot can have),
 - **demons** (procedural attachments such as if_needed, if_deleted, if_added etc.) and
 - **other** (may contain rules, other frames, semantic net or any type of other information).

Frame Network - Example





Detailed Representation of Frame Network



Description of Frames

- Each frame represents either a class or an instance.
- Class frame represents a general concept whereas instance frame represents a specific occurrence of the class instance.
- Class frame generally have default values which can be redefined at lower levels.
- If class frame has actual value facet then decedent frames can not modify that value.
- Value remains unchanged for subclasses and instances.

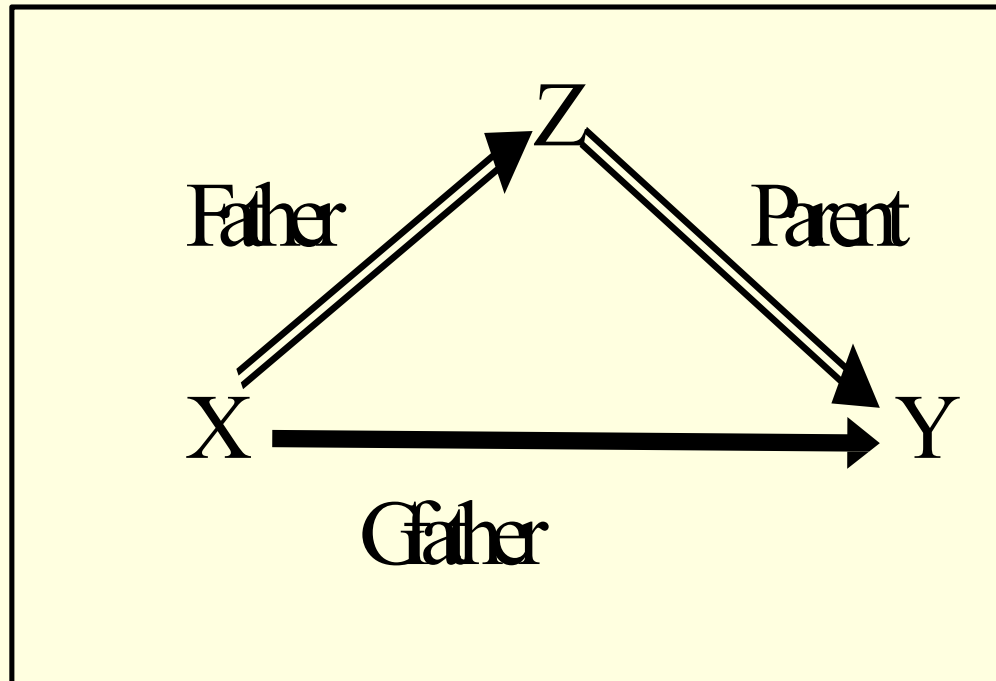
Extended Semantic Network

- In conventional Sem Net, clausal form of logic can not be expressed.
- Extended Semantic Network (ESNet) combines the advantages of both logic and semantic network.
- In the ESNet, terms are represented by nodes similar to Sem Net.
- Binary predicate symbols in clausal logic are represented by labels on arcs of ESNet.
 - An *atom* of the form “Love(john, mary)” is an arc labeled as ‘Love’ with its two end nodes representing ‘john’ and ‘mary’.
- *Conclusions* and *conditions* in clausal form are represented by different kinds of arcs.
 - Conditions are drawn with two lines  and conclusions are drawn with one heavy line .

Examples

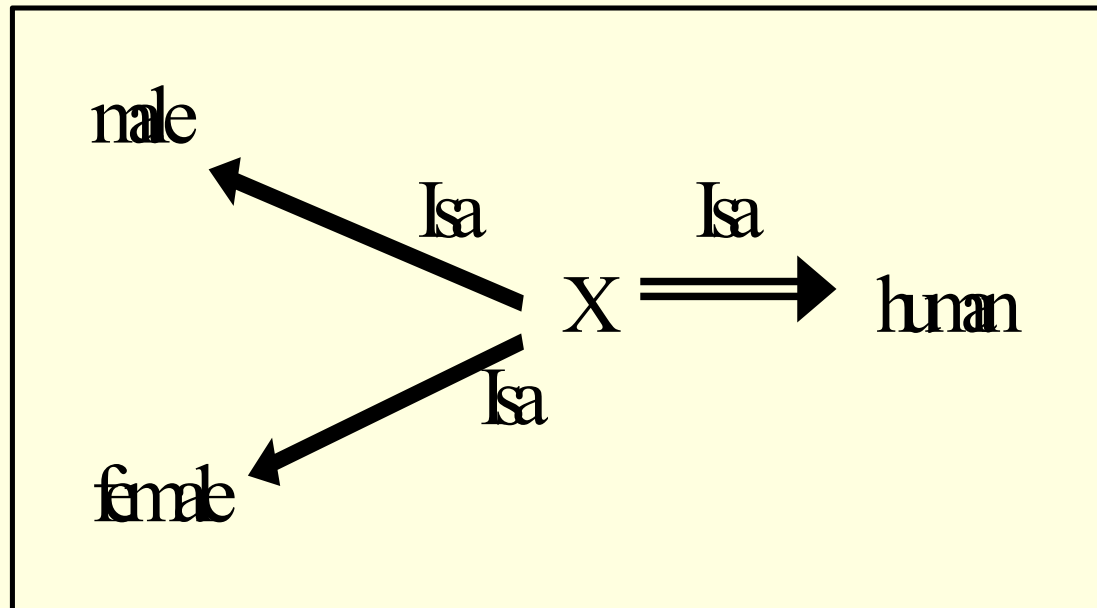
- Represent 'grandfather' definition

$\text{Gfather}(X, Y) \leftarrow \text{Father}(X, Z), \text{Parent}(Z, Y)$ in ESN_{et}.



Cont...Example

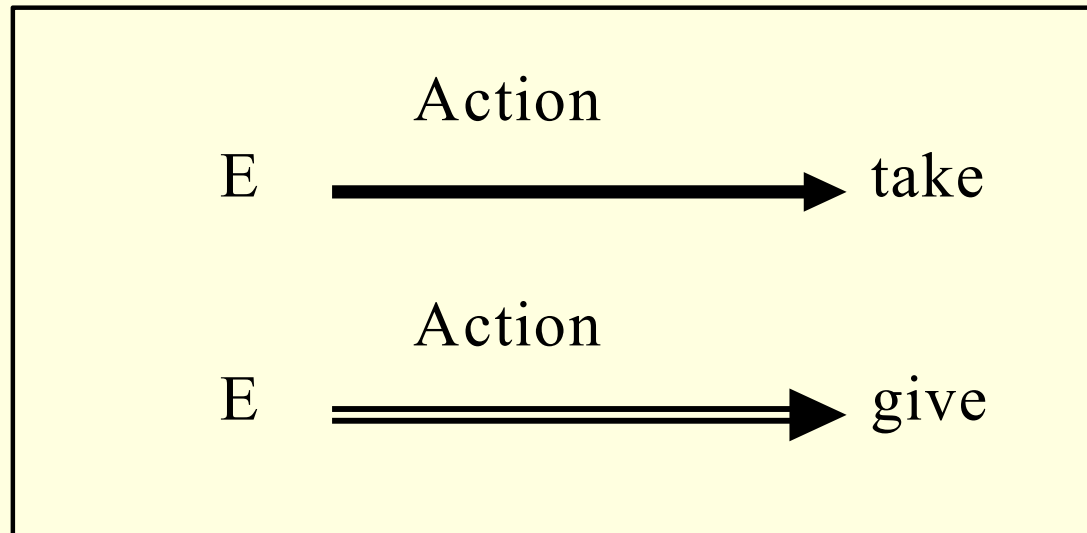
- Represent clausal rule “**Male(X), Female(X) \leftarrow Human(X)**” using binary representation as “**Isa(X, male), Isa(X, female) \leftarrow Isa(X, human)**” and subsequently in ESNet as follows:



Inference Rules in ESNet

- Inference rules are embedded in the representation itself.
- The inference that “for every action of giving, there is an action of taking” in clausal logic written as
“Action(E, take) \leftarrow Action(E, give)”.

ESNet

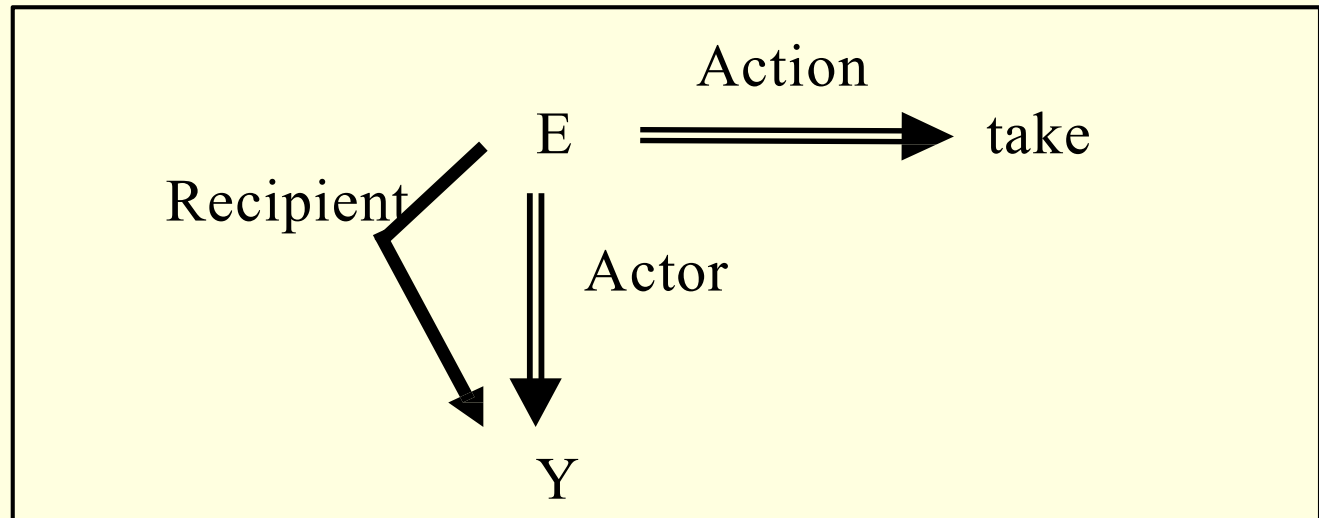


Cont...

- The inference rule such as “an actor of taking action is also the recipient of the action” can be easily represented in clausal logic as:
 - Here E is a variable representing an event where an action of taking is happening).

Recipient(E, Y) \leftarrow Acton(E, take), Actor (E, Y)

ESNet



Example

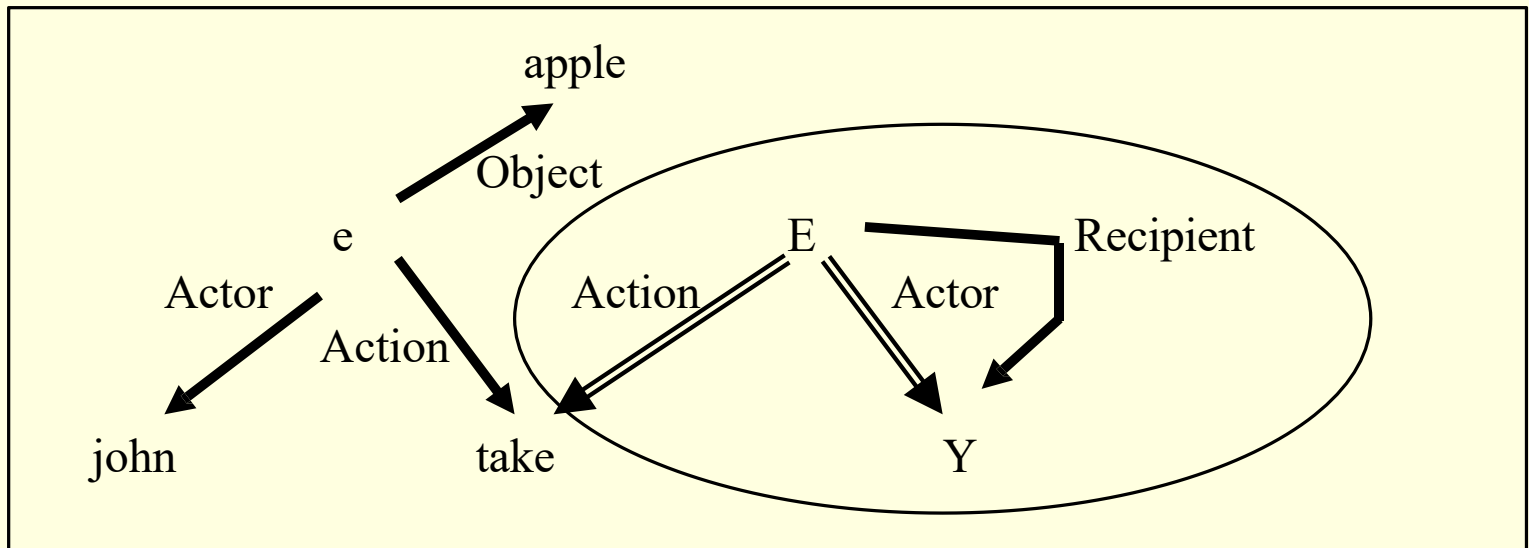
- Represent the following clauses of Logic in ESNet.

Recipient(E, Y) \leftarrow Acton(E, take), Actor (E, Y)

Object (e, apple).

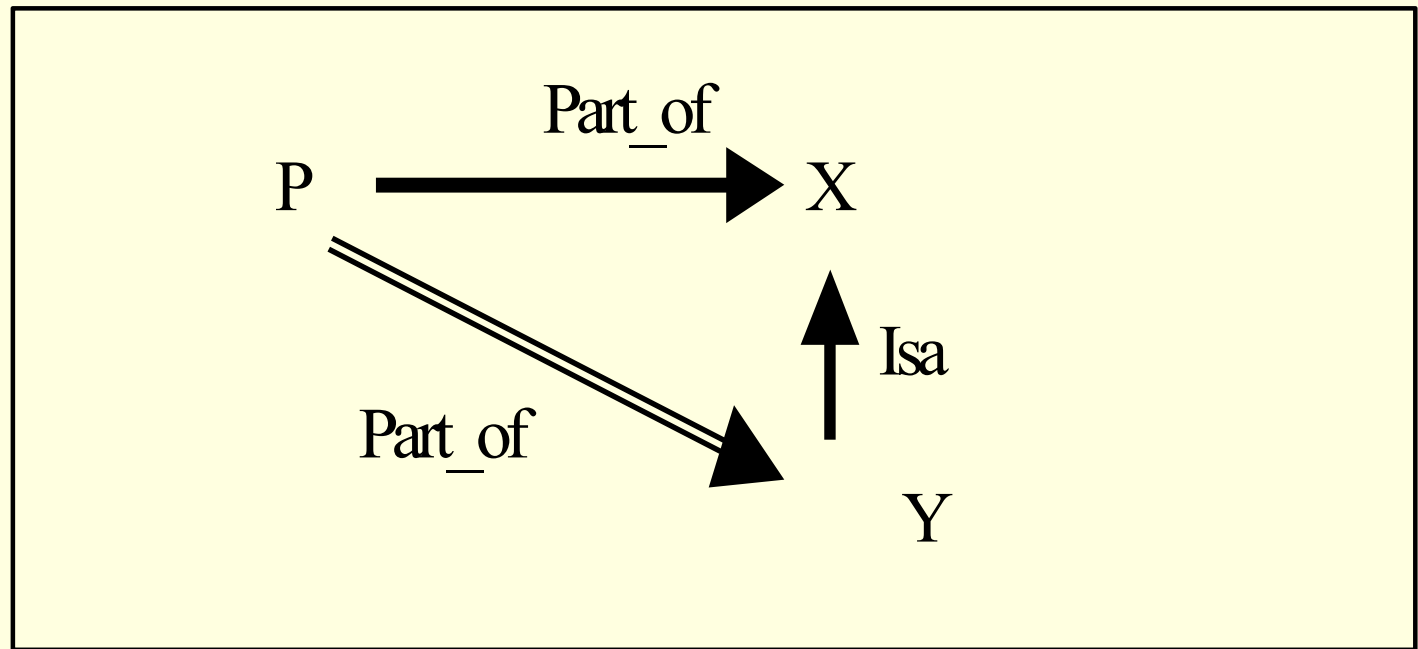
Action(e, take).

Actor (e, john) .



Contradiction

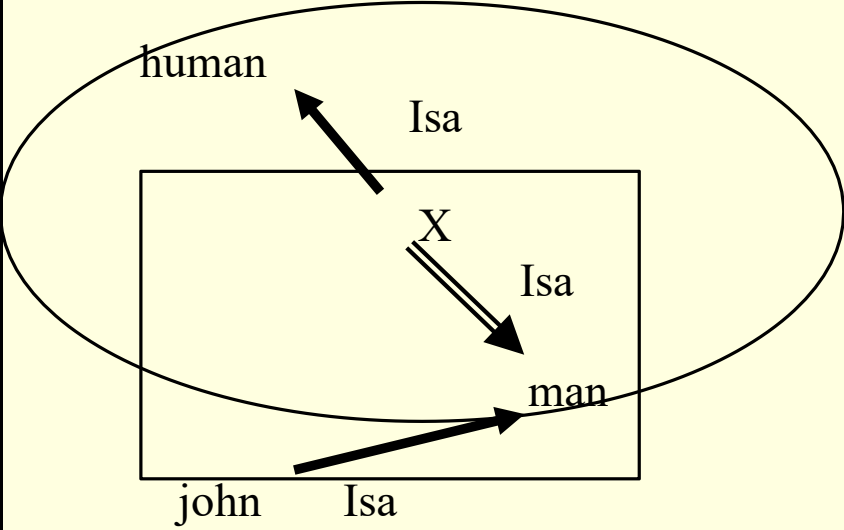
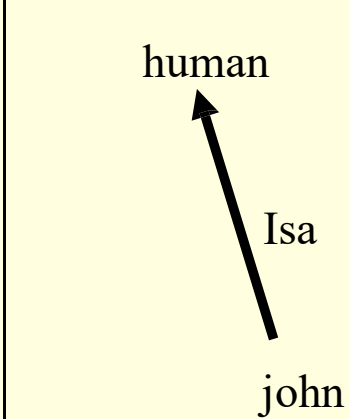
- The contradiction in the ESNet arises if we have the following situation.



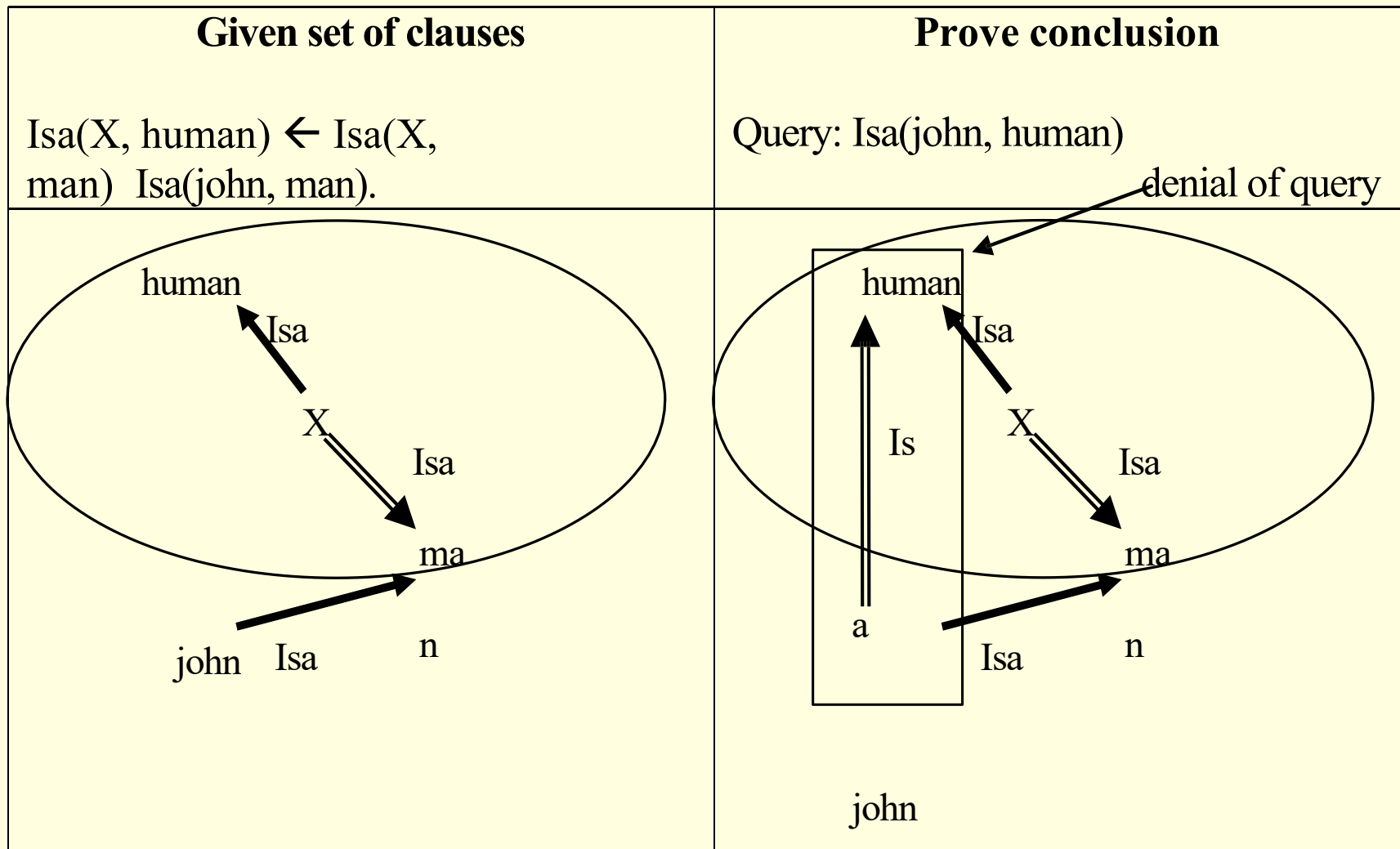
Deduction in ESNet

- Both of the following inference mechanisms are available in ESNet.
 - Forward reasoning inference (uses bottom up approach)
 - **Bottom Up Inferencing:** Given an ESNet, apply the following reduction (resolution) using modus ponens rule of logic ($\{A \leftarrow B, B\}$ then A).
 - Backward reasoning inference (uses top down approach).
 - **Top Down Inferencing:** Prove a conclusion from a given ESNet by adding the denial of the conclusion to the network and show that the resulting set of clauses in the network is inconsistent.

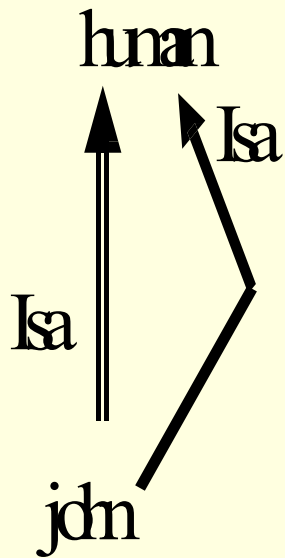
Example: Bottom Up Inferencing

Given set of clauses	Inferencing
<p>$\text{Isa}(X, \text{human}) \leftarrow \text{Isa}(X, \text{man})$ $\text{Isa}(\text{john}, \text{man}).$</p>	<p>$\text{Isa}(\text{john}, \text{human})$</p>
 <p>Here X is bound to john</p>	

Example: Top Down Inferencing



Cont...



$X = \text{john}$

Contradiction or Empty network is
generated Here 'Isa(john, human)'
is proved