10/9/2020

Module 1: Introduction to Artificial Intelligence and Knowledge based Systems

Course Title & Code: Principles of Artificial Intelligence/CSE 228

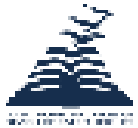
Semester: B.Tech

Department of Computer Science & Engineering,
School of Engineering, Presidency University,
Bengaluru

Contents

- Introduction to Artificial Intelligence, Definitions, foundation, History and Applications

Introduction to Artificial Intelligence



- Homo Sapiens : The name is Latin for *"wise man"*
- Philosophy of AI - *"Can a machine think and behave like humans do?"*
- In Simple Words -*Artificial Intelligence is a way of* ***making a computer, a computer-controlled robot, or a software think intelligently****, in the similar manner the intelligent humans think.*
- **Artificial intelligence** (**AI**) is an area of computer science that emphasizes the creation of **intelligent** machines that work and react like humans.
- AI is accomplished by studying how human brain thinks, and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intelligent software and systems.

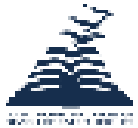
What is AI?

Views of AI fall into four categories:

1. Thinking humanly
2. Thinking rationally
3. Acting humanly
4. **Acting rationally**

The textbook advocates "acting rationally"

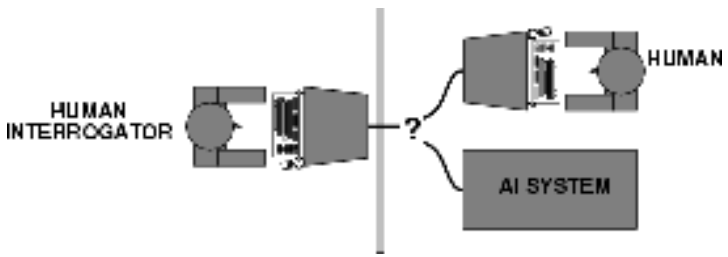
What is AI?



Thinking Humanly	Thinking Rationally
"The exciting new effort to make computers think … machines with minds, in the full and literal sense." "Activities that we associate with human thinking, activities such as decision-making, problem solving, learning…"	"The study of mental faculties through the use of computational models." "The study of the computations that make it possible to perceive, reason and act."
Acting Humanly	**Acting Rationally**
"The art of creating machines that perform functions that require intelligence when performed by people." "The study of how to make computers do things at which, at the moment, people are better."	"Computational Intelligence is the study of the design of intelligent agents." "AI … is concerned with intelligent behavior in artifacts."

Acting humanly: Turing Test

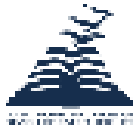
- Turing (1950) developed "Computing machinery and intelligence":
- "Can machines think?" □ "Can machines behave intelligently?"
- Operational test for intelligent behavior: **the Imitation Game**



A computer passes the test if a human interrogator, after posing some written questions, cannot tell whether the written responses come from a person or from a machine.

- Suggested major components of AI: knowledge, reasoning, language understanding, learning

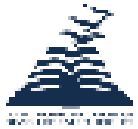
Acting humanly: Turing Test



The computer would need to posses the following capabilities:

- **Natural Language Processing:**
  To enable it to communicate successfully in English.
- **Knowledge representation:**
  To store what it knows or hear.
- **Automated reasoning:**
  To use the stored information to answer questions and to draw new conclusions.
- **Computer vision:**
  To perceive objects.
- **Robotics:**
  To manipulate objects and move about.

Thinking humanly: Cognitive Modeling

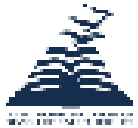


- If we are going to say that ***given program thinks like a human*** , we must have some way of determining how humans think.

- We need to get inside the actual working of human minds.

- There are 3 ways to do it:
    1. **Through introspection**
       Trying to catch our own thoughts as they go
    2. **Through psychological experiments**
       Observing a person in action
    3. **Through brain imaging** Observing the brain in action

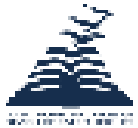
Thinking humanly: Cognitive Modeling



- Once we have a sufficiently precise theory of the mind, it becomes possible to express the theory as a computer program.

- If the program's input-output behavior matches corresponding human behavior, that is evidence that the program's mechanisms could also be working in humans

Thinking Rationally: “Laws of Thought"



- Aristotle: one of the first to attempt to codify **“right thinking”**. Mathematical representation.

- His **syllogisms** provided patterns for **argument structures** that always yielded correct conclusions when given premises are correct.

Example – Socrates is a man

All men are mortal

Therefore

Socrates is mortal

Acting Rationally: Rational Agent

- An agent is an entity that perceives and acts
- A system is rational if it does the “right thing,” given what it knows.
- This course is about designing rational agents
- Rational agent is one that acts so as to achieve the best outcome or, when there is uncertainty, the best expected outcome.

- Abstractly, an agent is a function from percept histories to actions:

$$[f : P^* \rightarrow A]$$

Behave Rationally……

- **What means "behave rationally" for a person/system:**
  - Take the right/ best action to achieve the goals, based on his/its knowledge and belief

Example: Assume I don't like to get wet in rain (my goal), so I bring an umbrella (my action). Do I behave rationally?

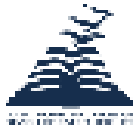
  - The answer is dependent on my knowledge and belief
  - If I've heard the forecast for rain and I believe it, then bringing the umbrella is rational.
  - If I've not heard the forecast for rain and I do not believe that it is going to rain, then bringing the umbrella is not rational

- **"Behave rationally" does not always achieve the goals successfully**

  Example:
    - My goals – (i) do not get wet if rain; (ii) do not looked stupid (such as bring an umbrella when not raining)
    - My knowledge/belief – weather forecast for rain and I believe it
    - My rational behaviour – bring an umbrella
    - The outcome of my behaviour: If rain, then my rational behaviour achieves both goals; If no rain, then my rational behaviour fails to achieve the 2nd goal

- **The successfulness of "behave rationally" is limited by my knowledge and belief**

Definition of AI



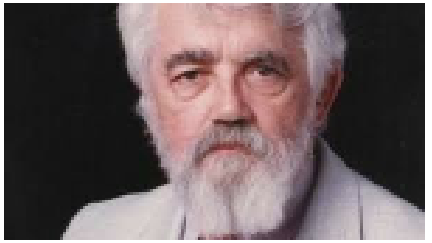
- Existing definitions advocate everything from replicating human intelligence to simply solving knowledge-intensive tasks.

Examples:

"Artificial Intelligence is the design, study and construction of computer programs that behave intelligently." -- Tom Dean.

"Artificial Intelligence is the enterprise of constructing a physical symbol system that can reliably pass the Turing test." -- Matt Ginsberg.

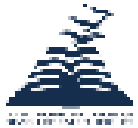
History of AI



- 1943 McCulloch & Pitts developed **Boolean circuit model of brain**
- 1950 Turing's **"Computing Machinery and Intelligence"**
- 1956 Dartmouth meeting: **"Artificial Intelligence"** adopted
- 1952—69 McCarthy referred **Look, Ma, no hands!** era People thought “Only arithmetic can be done and no more before 1952”. Astonishing if something is done remotely clever.
- 1950s Early AI programs, including
  - » **Samuel's checkers program**,
  - » **Newell & Simon's Logic Theorist** : It was the first program deliberately engineered to mimic the problem solving skills of a human being and is called "the first artificial intelligence program". It would eventually prove 38 of the first 52 theorems in Whitehead and Russell's *Principia Mathematica*
  - » **Gelernter's Geometry Engine**

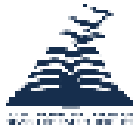
  Robinson's complete algorithm for logical reasoning
- 1965

AI Winter



- Funding Reduces
- P Vs NP
- Exponential time and complexity explosions
- Limited Computational Capabilities
  - RAM
  - Processing Power
  - Data
- Moravec's paradox@ CMU
  - High-level reasoning requires very little computation, but low-level subconscious tasks require enormous computational resources

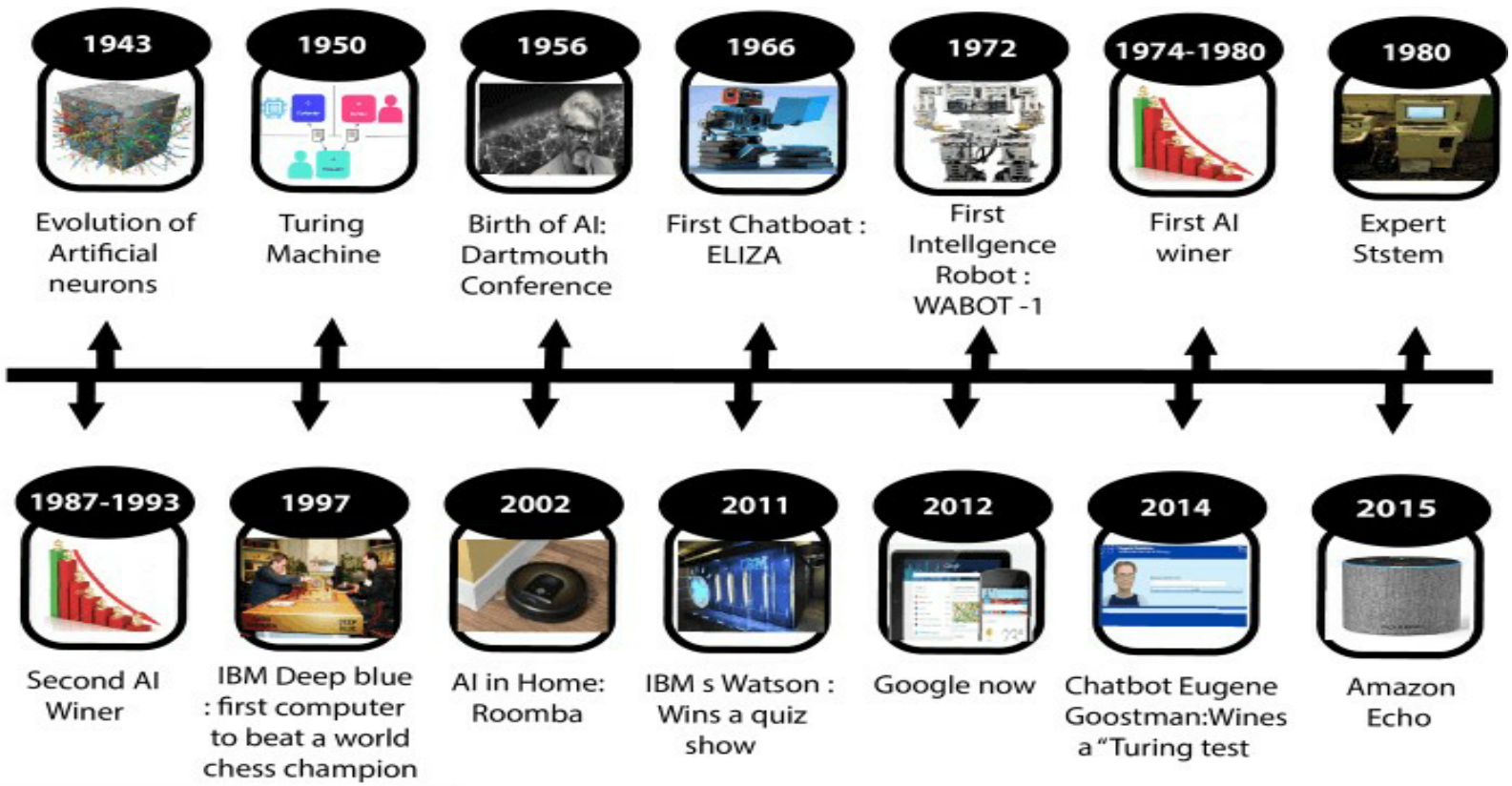
History of AI



- 1966—73 AI discovers **computational complexity** **Neural network research** almost **disappears**
- 1969—79 Early development of **knowledge-based systems**
- 1980-- AI becomes an **industry**
- 1986-- **Neural networks** return to popularity AI becomes
- 1987-- a **science**
- 1995-- The emergence of **intelligent agents**

10/9/2020

History of AI



Ref : https://www.javatpoint.com/history-of-artificial-intelligence

Foundations of Artificial Intelligence

Below are the disciplines that contributed **ideas**, **viewpoints** and **techniques** to AI:

1. Philosophy
2. Mathematics
3. Economics
4. Neuroscience
5. Psychology
6. Computer Engineering
7. Control theory
8. Linguistics

Foundations of Artificial Intelligence

- Philosophy
  - Can formal rules be used to draw valid conclusions?
  - How does the mind arise from a physical brain? Where does knowledge come from?
  - How does knowledge lead to action?

Aristotle was the first to formulate a precise set of laws governing the **rational part of the mind**. He developed an informal system of **syllogisms** for **proper reasoning**, which in principle allowed one to generate **conclusions** mechanically, given initial **premises**.

All dogs are animals; all animals have four legs; therefore all dogs have four legs

Descartes was a strong advocate of the **power of reasoning** in understanding the world, philosophy now called as **rationalism**.

Foundations of Artificial Intelligence

- Mathematics
  - What are the formal rules to draw valid conclusions? What can be computed?
  - How do we reason with uncertain information?

Formal representation and proof algorithms: Propositional logic
Computation: Turing tried to characterize exactly which functions are computable - capable of being computed.

(un)decidability: Incompleteness theory showed that in any formal theory, there are **true statements that are undecidable** i.e. they have no proof within the theory.

"a line can be extended infinitely in both directions"

(in)tractability: A problem is called intractable if the time required to solve instances of the problem grows exponentially with the size of the instance.
probability: Predicting the future.

Foundations of Artificial Intelligence

- **Economics**

  How should we make decisions so as to maximize payoff?

Economics is the study of how people make choices that lead to **preferred outcomes**(utility).

Decision theory: It combines **probability theory** with **utility theory**, provides a formal and complete framework for decisions made under uncertainty.

- **Neuroscience**

  How do brains process information?

Neuroscience is the study of the **nervous system**, particularly brain.
Brain consists of nerve cells or **neurons**. 10^11 neurons.
Neurons are considered as **Computational units**.

Foundations of Artificial Intelligence

- **Psychology**
  How do Humans and animals think and act?

- **Computer engineering**
  How can we build an efficient computer?

  Building fast computers

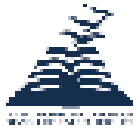
- **Control theory**
  How can artifacts operate under their own control?

  Design systems that maximize an objective function over time

- **Linguistics**
  How does the language relate to thought?

  knowledge representation, grammar

Applications of AI



Applications:

- **Deep Blue (chess-playing computer)** defeated the world chess champion Garry Kasparov in 1997.
- During the 1991 Gulf War, US forces deployed an **AI logistics planning and scheduling program** that involved up to 50,000 vehicles, cargo, and people
  Planning – How to use resources? Scheduling – When to use the resources?
- NASA's on-board autonomous planning program controlled the **scheduling of operations for a spacecraft**

- **Google duplex**
- The GPS developed in 1957 by Alan Newell and Hervert Simon, embodied a grandiose vision

Future Perspective

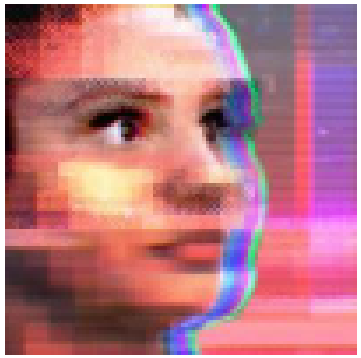
(1) Reducing the time and cost of development is a big plan for AI.

(2) To develop applications towards strong AI.

- (3) Allowing students to work collaboratively is another plan from Researchers.

- **Perfect rationality:** the classical notion of rationality in decision theory.
- **Bounded optimality:** A bounded optimal agent behaves as well as possible given its computational resources.
- **Game theory** studies decision problems in which the utility of a given action depends not only on chancing events in the environment but also on the actions of other agents.

Major Concerns



TAY

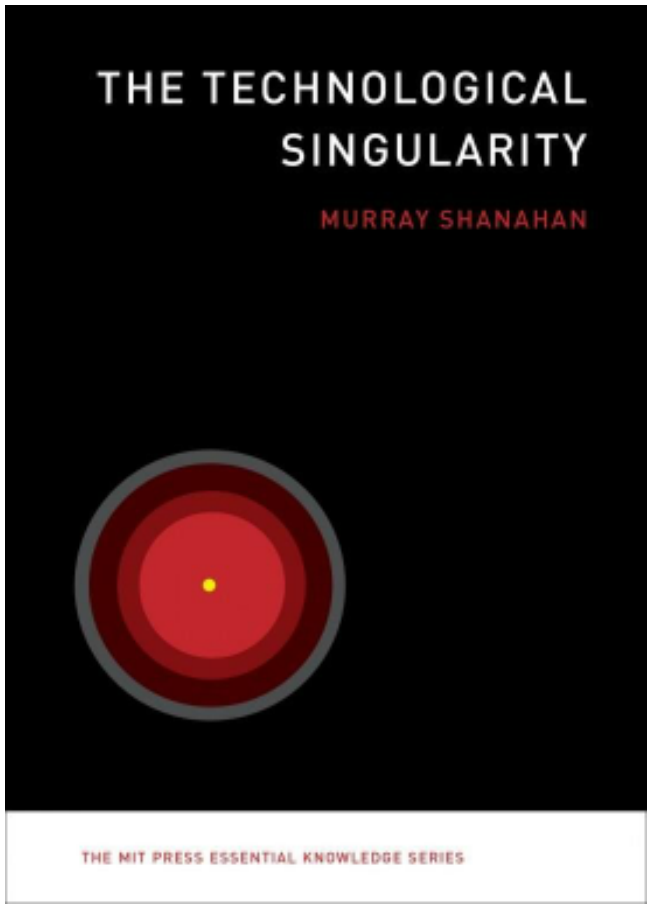
COMPAS - Correctional Offender Management Profiling for Alternative Sanctions

	HUMANS	COMPAS
Accuracy (overall)	67.0%	65.2%
False positive (black defendants)	37.1%	40.4%
False positive (white defendants)	27.2%	25.4%
False negative (black defendants)	29.2%	30.9%
False negative (white defendants)	40.3%	47.9%

10/9/2020

Singularity

Singularity is a hypothetical future point in time at which technological growth becomes uncontrollable and irreversible, resulting in unfathomable changes to human civilization



10/9/2020

- AGENTS

Agents in Artificial Intelligence

- Artificial intelligence is defined as a study of rational agents. A rational agent could be anything which makes decisions, as a person, firm, machine, or software. It carries out an action with the best outcome after considering past and current percepts(agent's perceptual inputs at a given instance).
  An AI system is composed of an **agent and its environment**. The agents act in their environment. The environment may contain other agents. An agent is anything that can be viewed as :
- perceiving its environment through **sensors** and
- acting upon that environment through **actuators**

The Structure of Intelligent Agents

- Agent’s structure can be viewed as −
- Agent = Architecture + Agent Program
- Architecture = the machinery that an agent executes on.
- Agent Program = an implementation of an agent function.

The Structure of Intelligent Agents

- To understand the structure of Intelligent Agents, we should be familiar with *Architecture* and *Agent Program* . **Architecture** is the machinery that the agent executes on. It is a device with sensors and actuators, for example : a robotic car, a camera, a PC. **Agent program** is an implementation of an agent function. An **agent function** is a map from the percept sequence(history of all that an agent has perceived till date) to an action.

Agent Terminology

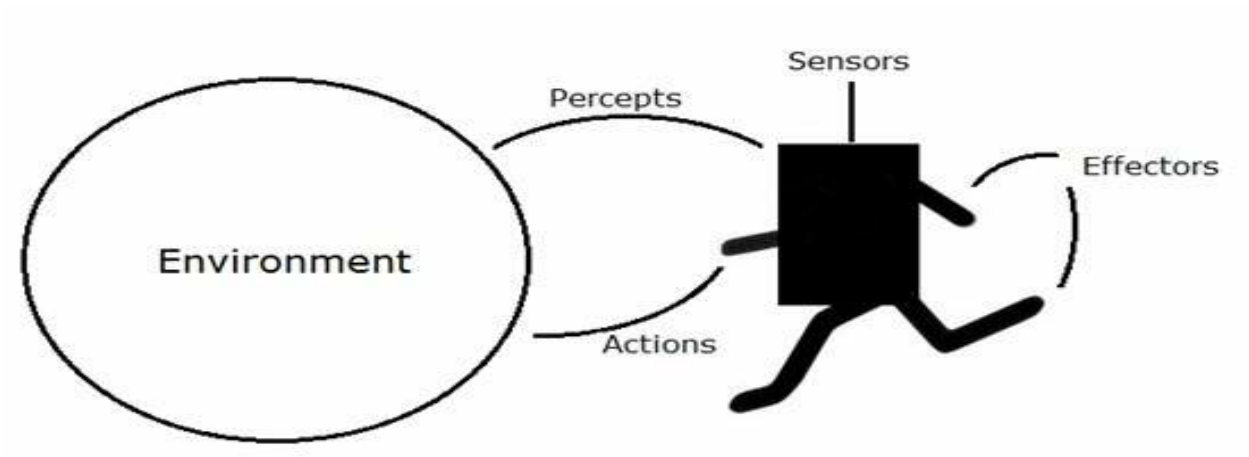
- **Performance Measure of Agent** − It is the criteria, which determines how successful an agent is.
- **Behavior of Agent** − It is the action that agent performs after any given sequence of percepts.
- **Percept** − It is agent's perceptual inputs at a given instance.
- **Percept Sequence** − It is the history of all that an agent has perceived till date.
- **Agent Function** − It is a map from the precept sequence to an action.

Rationality

- Rationality is nothing but status of being reasonable, sensible, and having good sense of judgment.
- Rationality is concerned with expected actions and results depending upon what the agent has perceived. Performing actions with the aim of obtaining useful information is an important part of rationality.
- **What is Ideal Rational Agent?**
- An ideal rational agent is the one, which is capable of doing expected actions to maximize its performance measure, on the basis of −
- Its percept sequence
- Its built-in knowledge base
- Rationality of an agent depends on the following −
- The **performance measures**, which determine the degree of success.
- Agent's **Percept Sequence** till now.
- The agent's **prior knowledge about the environment**.
- The **actions** that the agent can carry out.
- A rational agent always performs right action, where the right action means the action that causes the agent to be most successful in the given percept sequence. The problem the agent solves is characterized by Performance Measure, Environment, Actuators, and Sensors (PEAS).

Examples of Agent:-

- An **agent** is anything that can perceive its environment through **sensors** and acts upon that environment through **effectors.**
- A **human agent** has sensory organs such as eyes, ears, nose, tongue and skin parallel to the sensors, and other organs such as hands, legs, mouth, for effectors.
- A **robotic agent** replaces cameras and infrared range finders for the sensors, and various motors and actuators for effectors.
- A **software agent** has encoded bit strings as its programs and actions.



Types of Agents

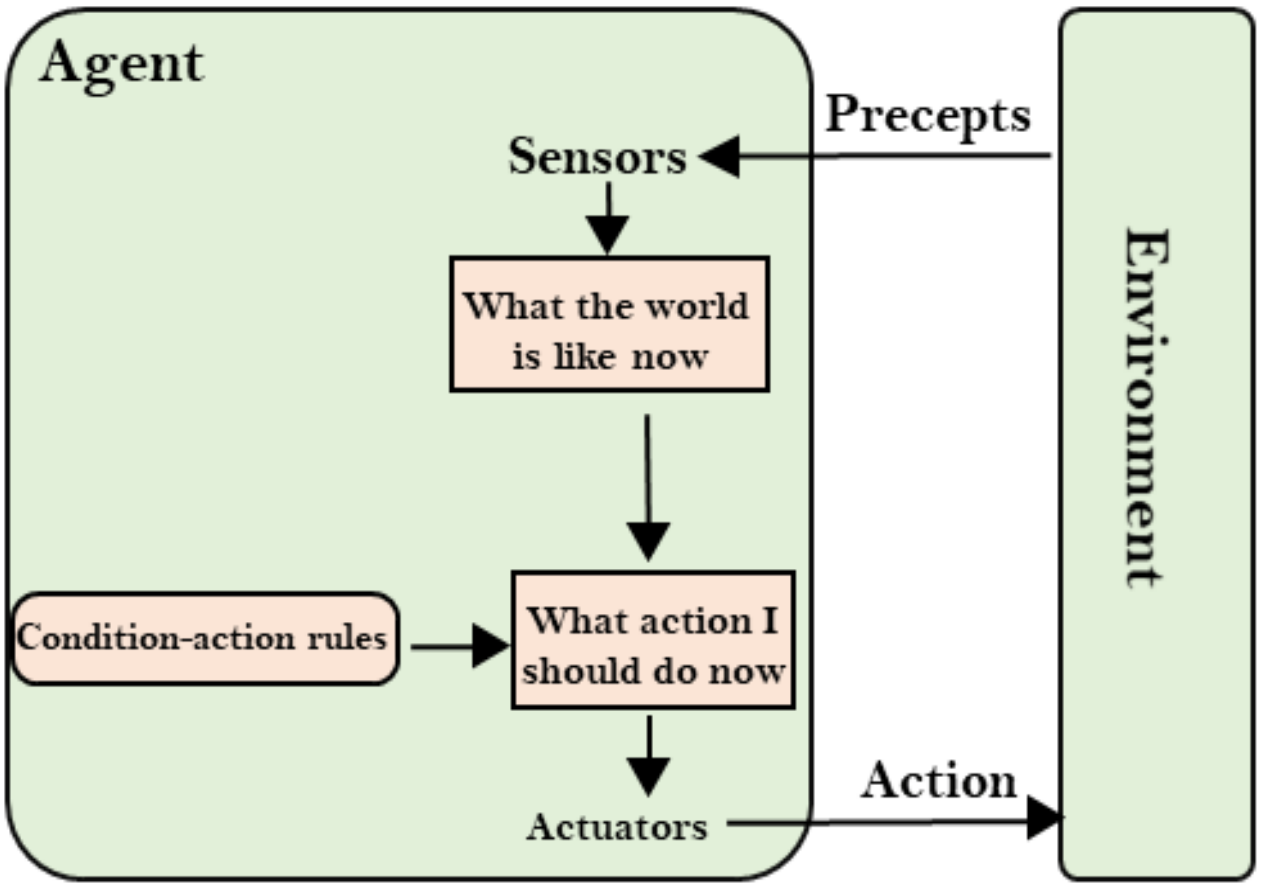
- Agents can be grouped into five classes based on their degree of perceived intelligence and capability. All these agents can improve their performance and generate better action over the time. These are given below:
- Simple Reflex Agent
- Model-based reflex agent
- Goal-based agents
- Utility-based agent
- Learning agent

Simplex Agent

- The Simple reflex agents are the simplest agents. These agents take decisions on the basis of the current percepts and ignore the rest of the percept history.
- These agents only succeed in the fully observable environment.
- The Simple reflex agent does not consider any part of percepts history during their decision and action process.
- The Simple reflex agent works on Condition-action rule, which means it maps the current state to action. Such as a Room Cleaner agent, it works only if there is dirt in the room.
- Problems for the simple reflex agent design approach:
  - They have very limited intelligence
  - They do not have knowledge of non-perceptual parts of the current state
  - Mostly too big to generate and to store.
  - Not adaptive to changes in the environment.

1
0
/
9
/
2
0
2
0

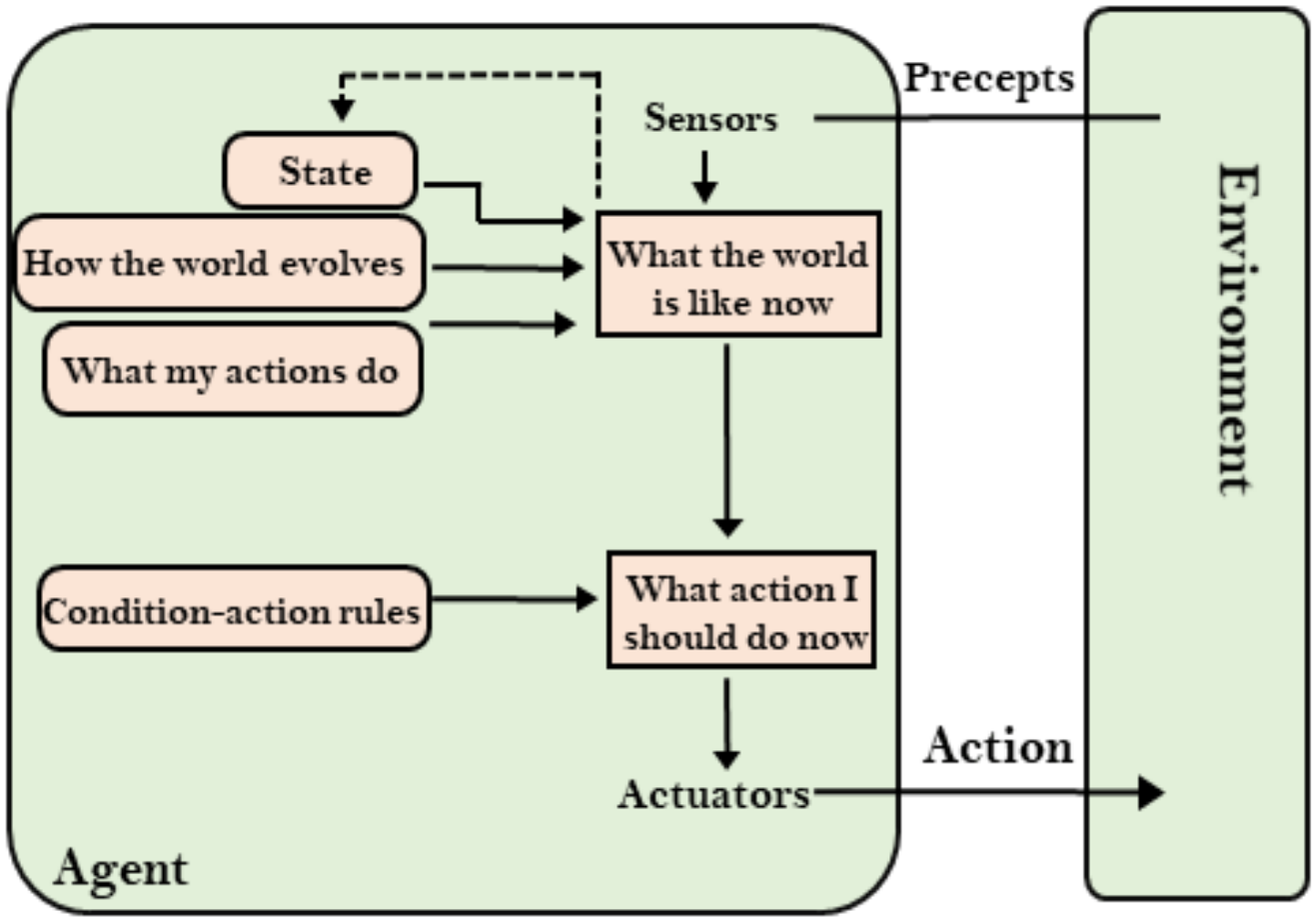
Simplex Agent



Model-based reflex agent

- The Model-based agent can work in a partially observable environment, and track the situation.
- A model-based agent has two important factors:
  - **Model:** It is knowledge about "how things happen in the world," so it is called a Model-based agent.
  - **Internal State:** It is a representation of the current state based on percept history.
- These agents have the model, "which is knowledge of the world" and based on the model they perform actions.
- Updating the agent state requires information about:
  - How the world evolves
  - How the agent's action affects the world.

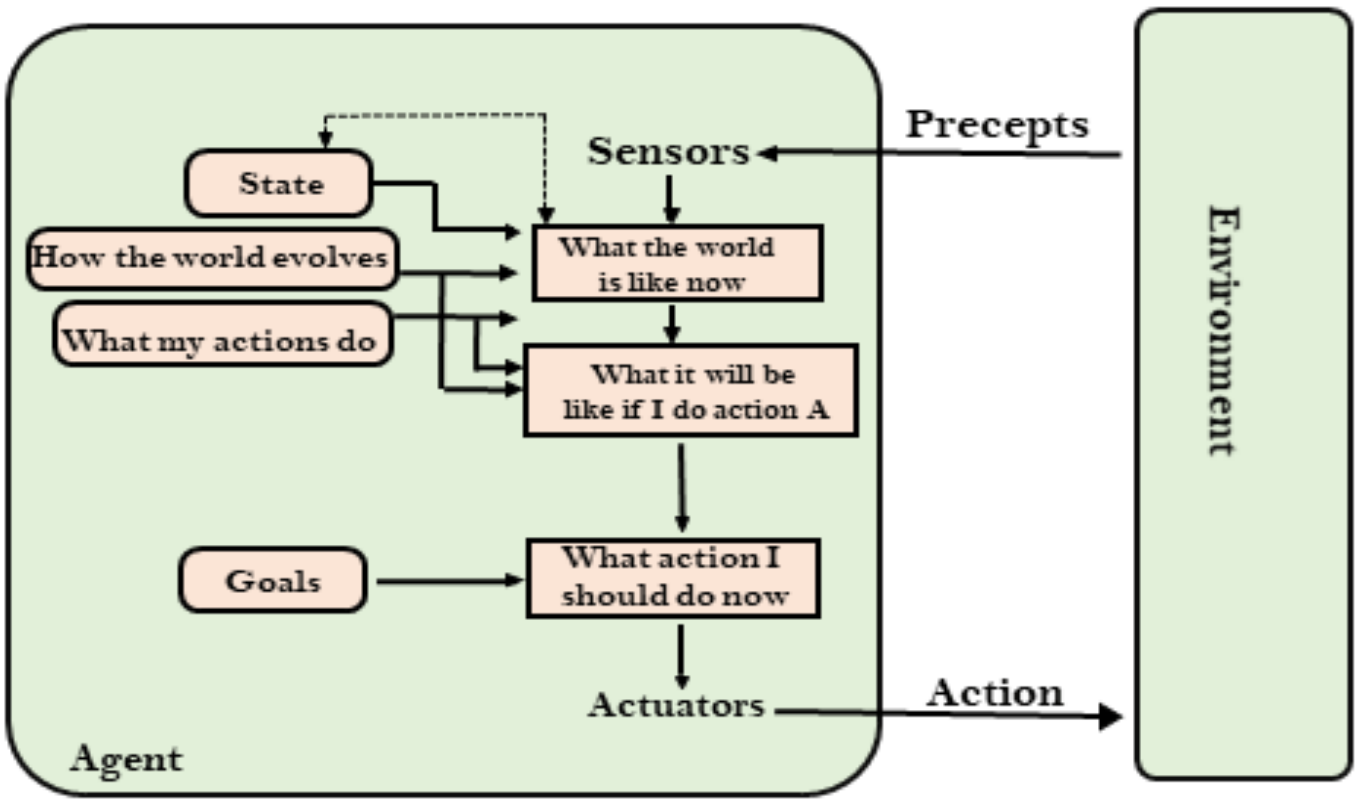
Model-based reflex agent



Goal-based agents

- The knowledge of the current state environment is not always sufficient to decide for an agent to what to do.
- The agent needs to know its goal which describes desirable situations.
- Goal-based agents expand the capabilities of the model-based agent by having the "goal" information.
- They choose an action, so that they can achieve the goal.
- These agents may have to consider a long sequence of possible actions before deciding whether the goal is achieved or not. Such considerations of different scenario are called searching and planning, which makes an agent proactive.

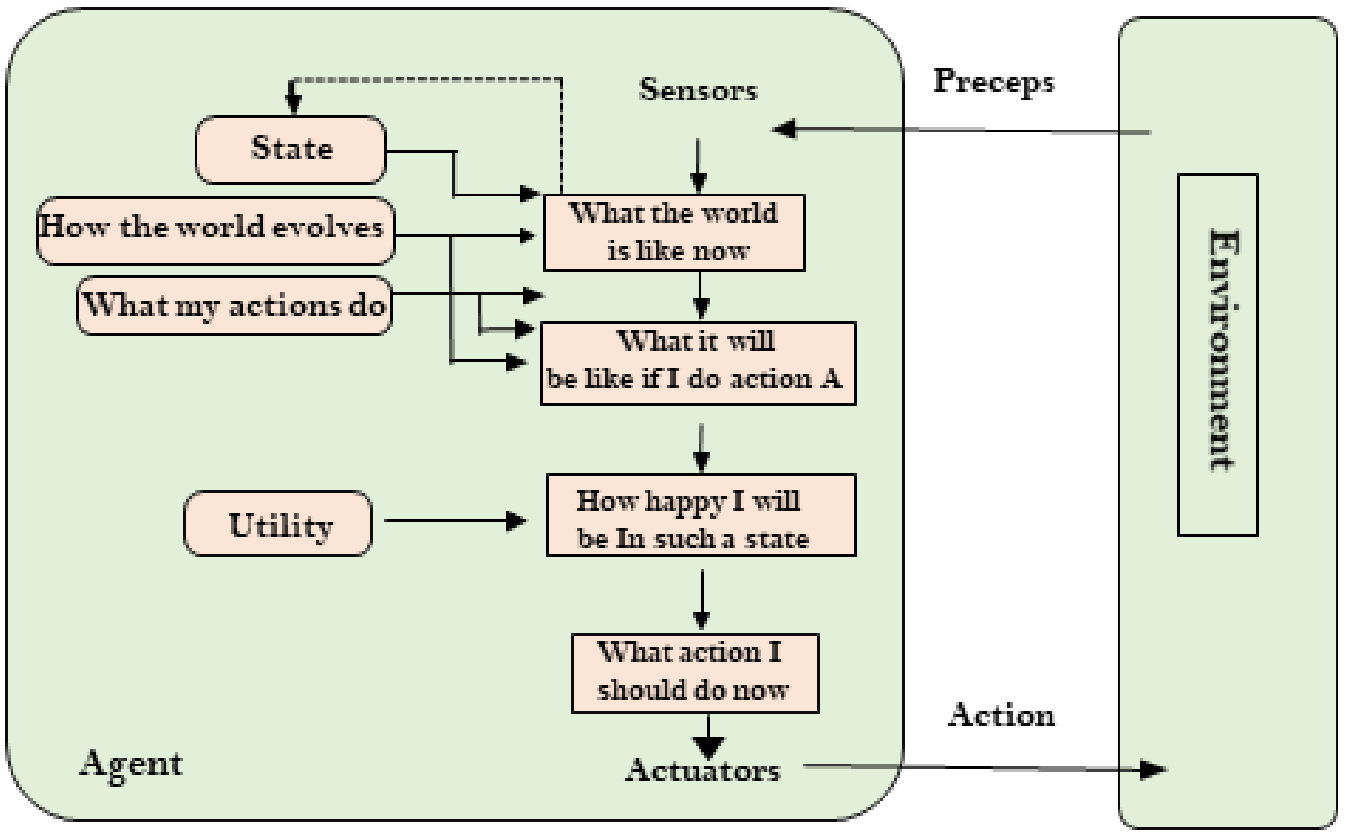
Goal-based agents



Utility-based agents

- These agents are similar to the goal-based agent but provide an extra component of utility measurement which makes them different by providing a measure of success at a given state.
- Utility-based agent act based not only goals but also the best way to achieve the goal.
- The Utility-based agent is useful when there are multiple possible alternatives, and an agent has to choose in order to perform the best action.
- The utility function maps each state to a real number to check how efficiently each action achieves the goals.

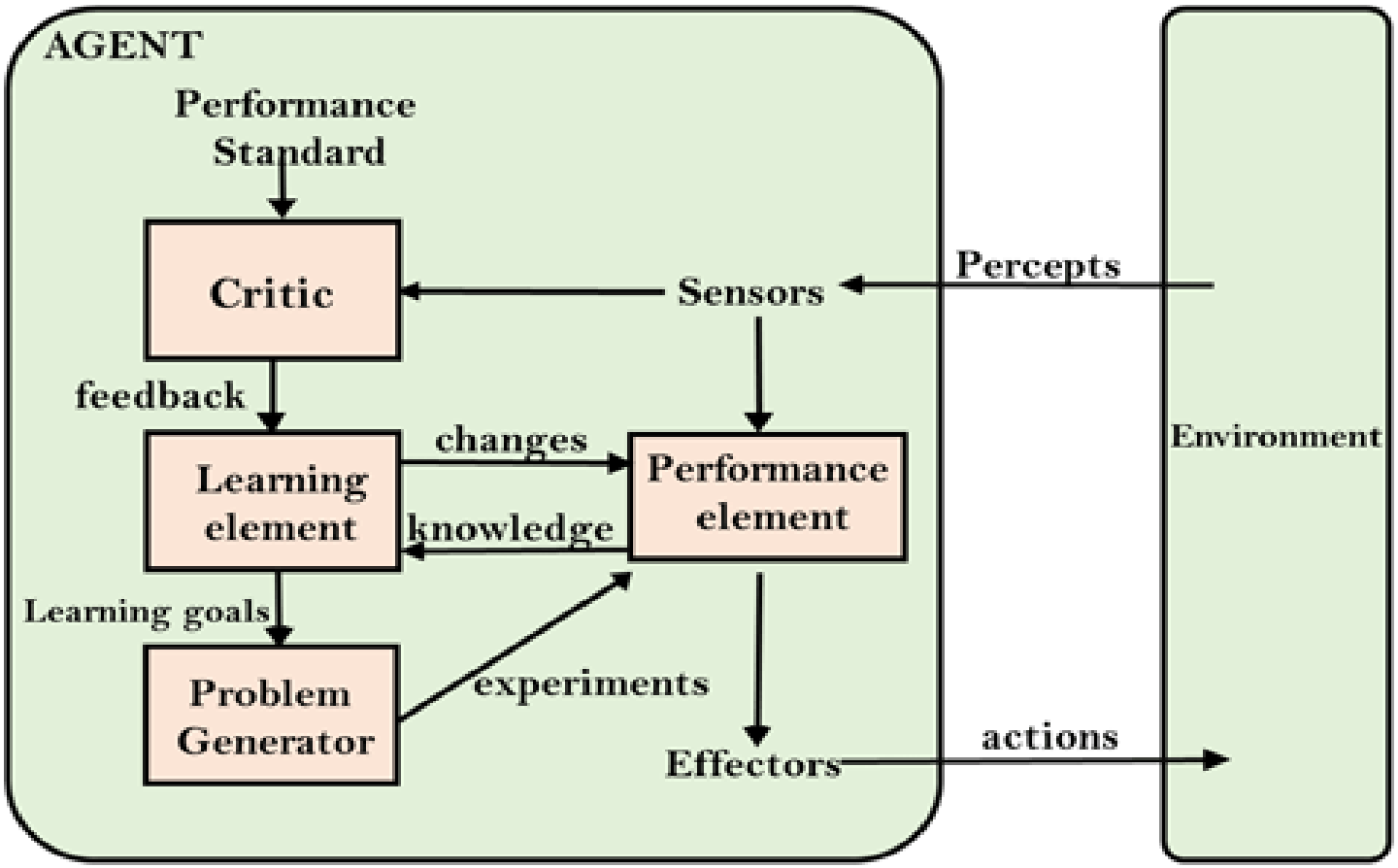
Utility-based agents



Learning Agents

- A learning agent in AI is the type of agent which can learn from its past experiences, or it has learning capabilities.
- It starts to act with basic knowledge and then able to act and adapt automatically through learning.
- A learning agent has mainly four conceptual components, which are:
  - **Learning element:** It is responsible for making improvements by learning from environment
  - **Critic:** Learning element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard.
  - **Performance element:** It is responsible for selecting external action
  - **Problem generator:** This component is responsible for suggesting actions that will lead to new and informative experiences.
- Hence, learning agents are able to learn, analyze performance, and look for new ways to improve the performance.

Learning Agents



10/9/2020

- KNOWLEDGE

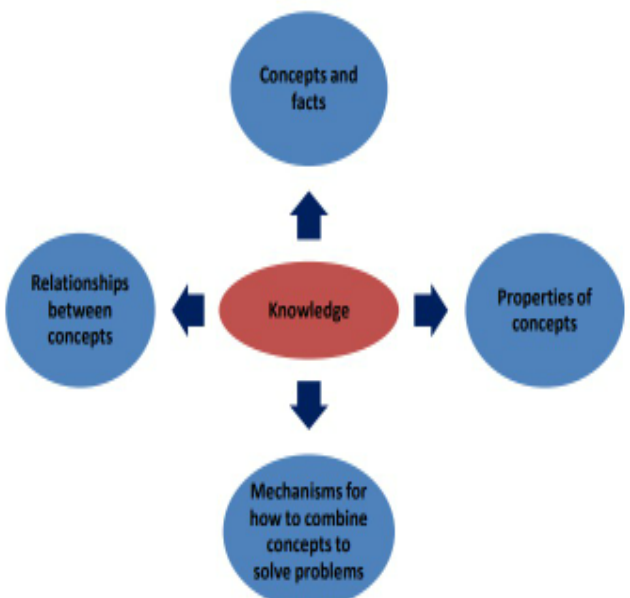
Knowledge

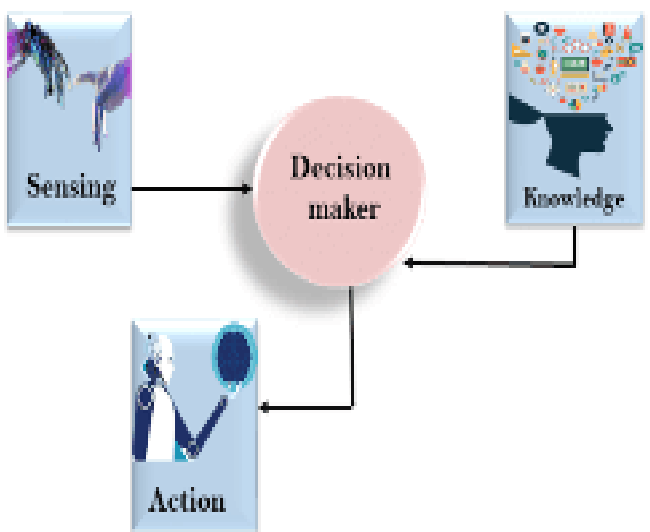
- Definition and Importance of Knowledge
- Knowledge-Based Systems
- Knowledge Organization
- Representation of Knowledge
  - Logic
  - Associative Networks
  - Frame Structures
  - Conceptual graphs

What is Knowledge ?

- Knowledge is the sort of information that people use to solve problems.
- **Knowledge is having familiarity with the language, concepts, procedures, rules, ideas, places, customs, facts, and associations.**

- Knowledge is understanding of a subject area.





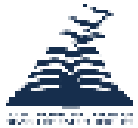
Ref : https://nptel.ac.in/courses/126104006/LectureNotes/Week-3_Knowledge%20Representation.pdf https://www.javatpoint.com/knowledge-representation-in-ai

What is Knowledge?

- Difference between data, information and knowledge:
  - Data: Primitive verifiable facts. Example: name of novels available in a library.
  - Information: Analyzed data. Example: The novel that is frequently asked by the members of library is "Harry Potter and the Chamber of Secrets".
  - Knowledge: Analyzed information that is often used for further information deduction. Example: Since the librarian knows the name of the novel that is frequently asked by members, s/he will ask for more copies of the novel the next time s/he places an order.

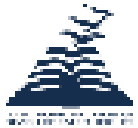
Ref : https://nptel.ac.in/courses/126104006/LectureNotes/Week-3_Knowledge%20Representation.pdf

Importance of Knowledge



- Knowledge is Power
- Knowledge is the primary factor that distinguishes the human race from animals
- knowledge is even what prevents us from making the same mistakes we made in the past.
- With knowledge, one can improve their abilities of thinking critically
- To make program intelligent provide it with lots of high quality specific knowledge about some problem area.
- AI systems are divided into a Knowledge Base with facts about the world and rules and an inference engine that applies the rules to the knowledge base in order to answer questions and solve problems.

Importance of Knowledge



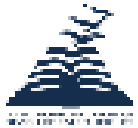
- Common way to represent knowledge is in the form of

  written language.

- For example, some facts and relations represented in English are:

1. **Joe is tall.**
2. **Bill likes Sue.**
3. **Sam has learned to use recursion to manipulate linked list in several programming languages.**

Importance of Knowledge



Joe is tall.

- knowledge above expresses simple fact.
- an attribute possessed by a person.

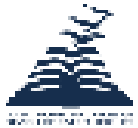
Bill likes Sue.

- knowledge above expresses complex binary relation between two persons.

Sam has learnt to use recursion to manipulate linked list in several programming languages.

- knowledge above is the most complex.
- Expressing relations between a person and more abstract programming concepts.

Important points on Knowledge



Knowledge may be declarative or procedural

- **Procedural Knowledge**

  - It is a compiled knowledge related to the performance of some task.

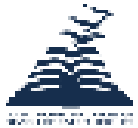
  For example, the steps used to solve an algebraic equations are expressed as procedural knowledge.

- **Declarative Knowledge**

  - Passive knowledge expressed as statements of facts about the world.

  For example, personnel data in a database.

Important points on Knowledge



Knowledge should not be confused with data

Example:

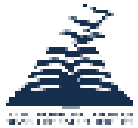
A physician treating a patient uses both knowledge and data.

Data: Patient's record, including patient history, drugs given, responses to drugs, and so on.

Knowledge: What the physician has learnt in medical school and in the years of internship, specialization and practice.

It consists of facts, prejudices and beliefs.

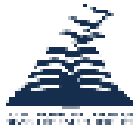
Important points on Knowledge



- Knowledge includes and requires the use of data and information.
- It combines relationships, correlations and dependencies.

10/9/2020

Important points on Knowledge



Distinction between Knowledge and other concepts such as belief and hypotheses

Belief: Any meaningful and coherent expression that can be represented. It may be true or false.

E.g. It is not good to have lunch after 4.30.

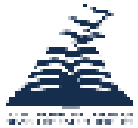
Hypotheses: A Justified belief that is not known to be true.

E.g. Tomorrow guests are coming; so prepare Fried Rice for lunch. They may like it.

Knowledge: True justified belief.

E.g. If you want to get better marks, you need to practice more. By practicing more, you can remember the concepts.

Types Of Knowledge



1. **Procedural Knowledge** - describes *how* a problem is solved
   - provides direction on how to do something
   - Ex: rules, strategies, agendas and procedures
2. **Declarative Knowledge** - describes *what* is known about a problem
   - includes simple statements that are asserted to be true or false
   - includes a list of statements that more fully describe some concept
3. **Meta-Knowledge** - describes *knowledge about knowledge*
   - used to pick other knowledge best suited for solving a problem
   - experts use to enhance the efficiency of problem solving by directing their reasoning into the most promising areas

Types Of Knowledge

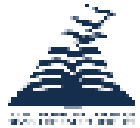
4. **Heuristic Knowledge** - describes a *rule-of-thumb* that guides the reasoning process.
   - represents the knowledge compiled by an expert through the experience of solving past problems
   - experts will take fundamental knowledge and compile it into simple heuristics to aid problem solving

5. **Structural Knowledge** - describes knowledge *structures*
   - describes the expert's overall mental model of the problem
   - concepts, sub-concepts, and objects are typical examples

Types of Knowledge

Procedural Knowledge	Rules; Strategies Agendas; Procedures
Declarative Knowledge	Concepts; Objects Facts
Meta-Knowledge	Knowledge about the other types of knowledge and how to use them
Heuristic Knowledge	Rules of Thumb
Structural Knowledge	Rule Sets; Concept Relationships Concept to Object Relationships

Knowledge Based Systems (KBS)

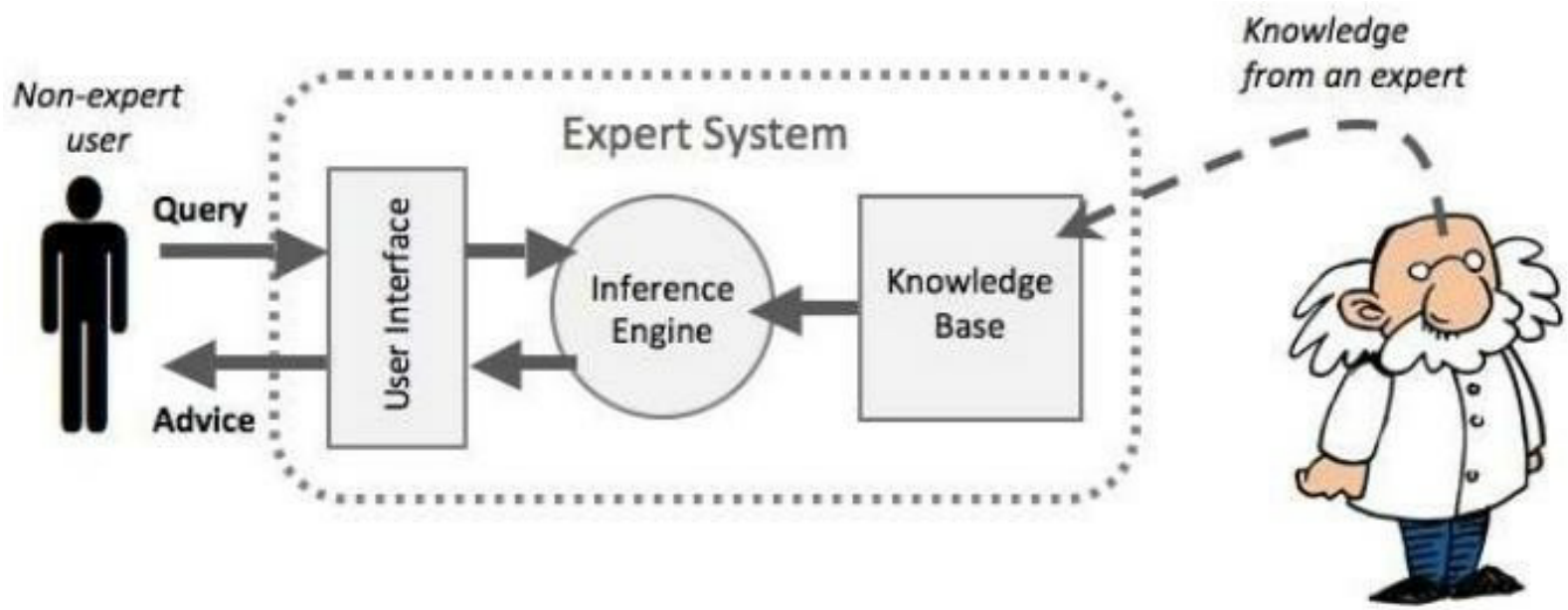


- A knowledge-based system (KBS) is a computer program that reasons and uses a knowledge base to solve complex problems
- A system which is built around a knowledge base. i.e. a collection of knowledge, taken from a human, and stored in such a way that the system can reason with it
- Uses AI to solve problems within a specialized domain that ordinarily requires human expertise
- Uses Heuristic (cause and effect) rather than algorithms
- E.g.
  - Expert Systems
  - Clinical decision-support systems
    - MYCIN, for example, was an early knowledge-based system created to help doctors diagnose diseases

KBS Examples

- **Expert Systems**
  - One in which the knowledge, stored in the knowledge base, has been taken from an expert in some particular field
  - **Expert systems** are designed to solve complex problems by reasoning through bodies of knowledge, represented mainly as if–then rules rather than through conventional procedural code
  - Therefore, an expert system can, to a certain extent, act as a substitute for the expert from whom the knowledge was taken
- **Clinical decision-support systems**
    - MYCIN, for example, was an early knowledge-based system created to help doctors diagnose diseases

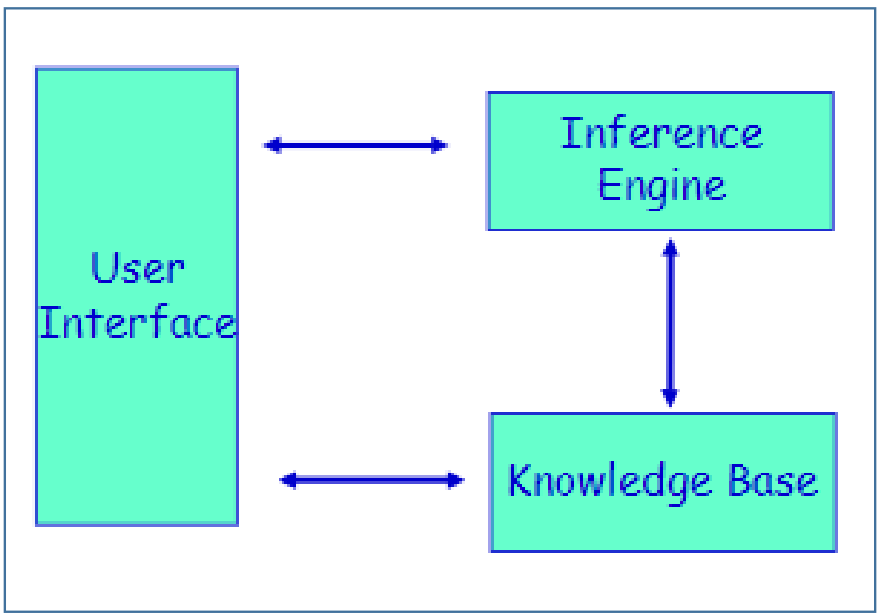
KBS Architecture



https://www.researchgate.net/figure/Expert-System-Architecture-23-Knowledge-Base_fig1_319208444

KBS Architecture

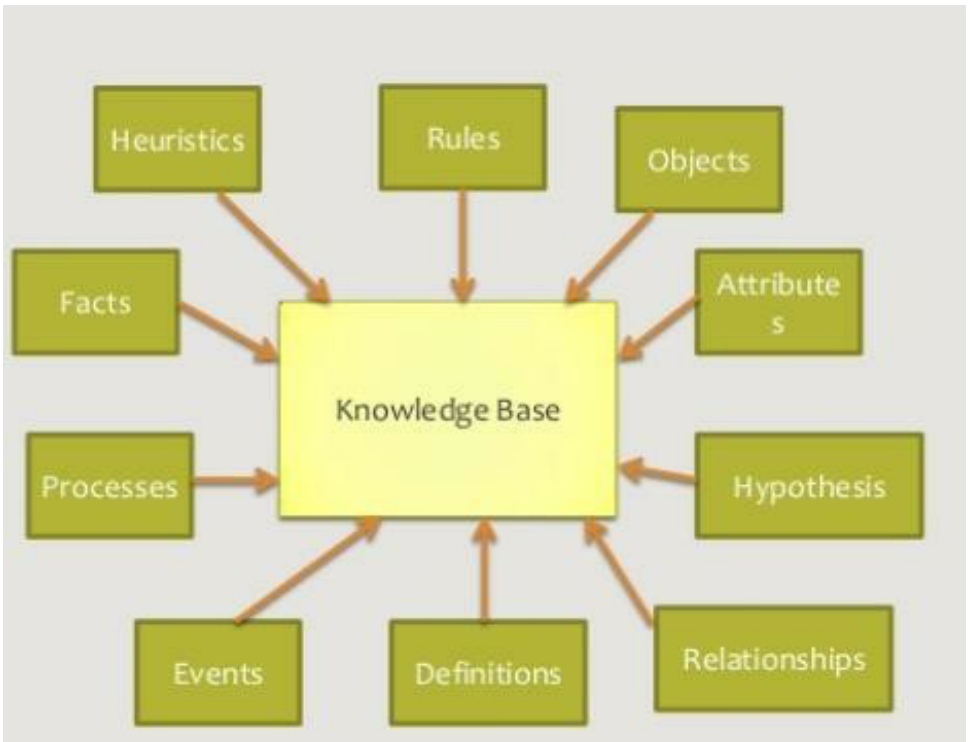
- **User Interface**
  - Enables the user to communicate with KBS



KBS Architecture

KBS = Knowledge-Base + Inference Engine

- **Knowledge Base** – The component of KBS that contains the system's knowledge in organized collection of facts about the system's domain



Knowledge base System

Storing knowledge inside the program

```
#include<stdio.h>
int main()
{
   char dob="20/08/1992";
   ……..
   ……..
}
```

#Knowledge

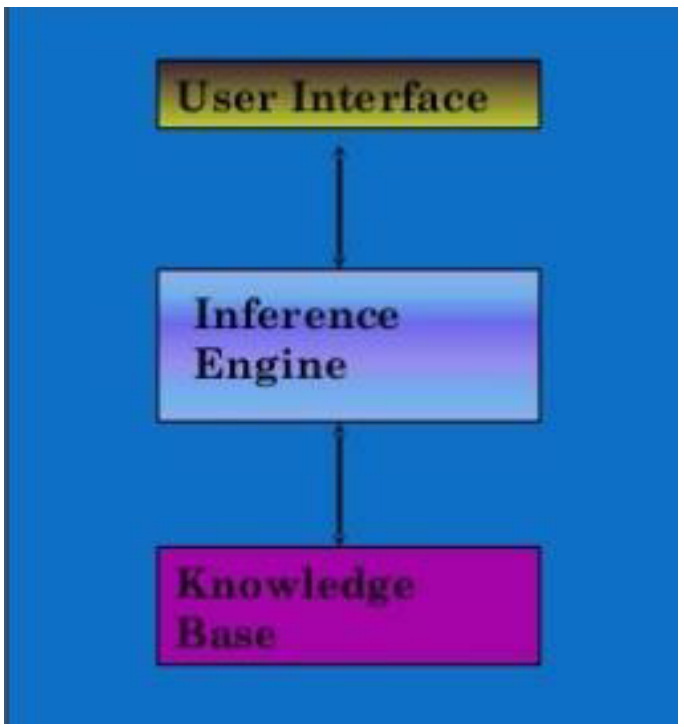
Instead write the dob in text file and access the date of birth from the text file

20/08/1992

Text file: dob.
txt

KBS Architecture

- **Inference Engine :**
  - Tries to derive answers from knowledge base
  - Brain of KBS that provides a methodology for reasoning about the information in the knowledge base and for formulating conclusions



Example 1 for AI system:
Gender Identification Problem

- **Male and female names have some distinctive characteristics.**
- **Names ending in *a*, *e* and *i* are likely to be female.**
- **Names ending in *k*, *n*, *r*, *s* and *t* are likely to be male.**

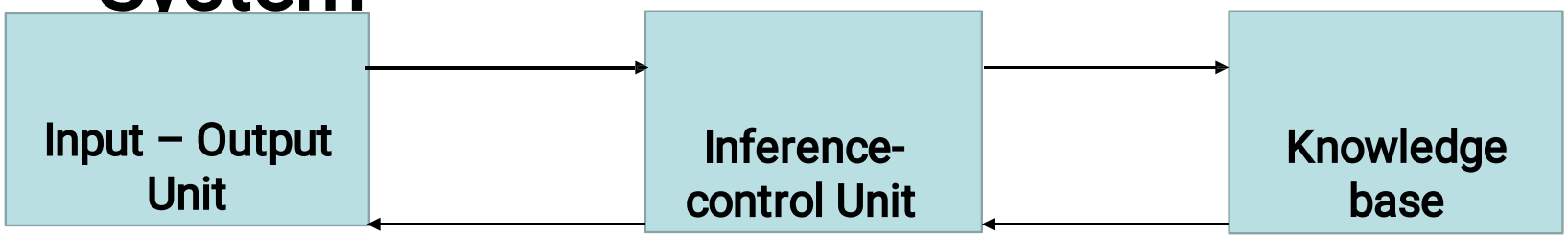
Input File: male.txt
Amit Prasan
Ashok Ankit Amar
Chetan Shashank
Sumant

Input File: female.txt
Reshma Akshata
Vani Sita Bhavani
Lalita Ankita Harika

Output: Predict the Gender for the following names:

- Karan
- Sameera

Architecture of AI
Components of Knowledge base System



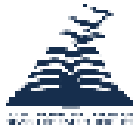
Input-output: male.txt and female.txt

Knowledge base:

From the last character in the name identification of gender is possible.

Inference-control Unit: AI Algorithm Implementation – Programs

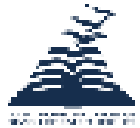
Example: Naïve Bayes Algorithm, Decision Tree Algorithm



List of Common Algorithms:

- Naive Bayes
- Decision Trees
- Linear Regression
- Support Vector Machines (SVM)
- Neural Networks

Example 2 for AI system:



Movie Rating

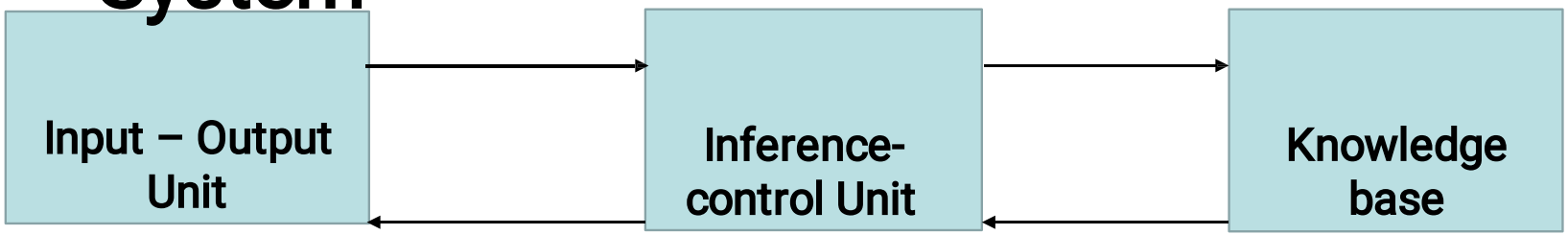
Airlift Movie rating – Reviews (Movie site, Facebook, Blog)

Post - How is Airlift Movie?

Comments from the people who watched movie -

- Airlift Movie is nice.
- It's boring.
- Yesterday I went to the movie. I enjoyed it.
- Superb.

Architecture of AI
Components of Knowledge base System



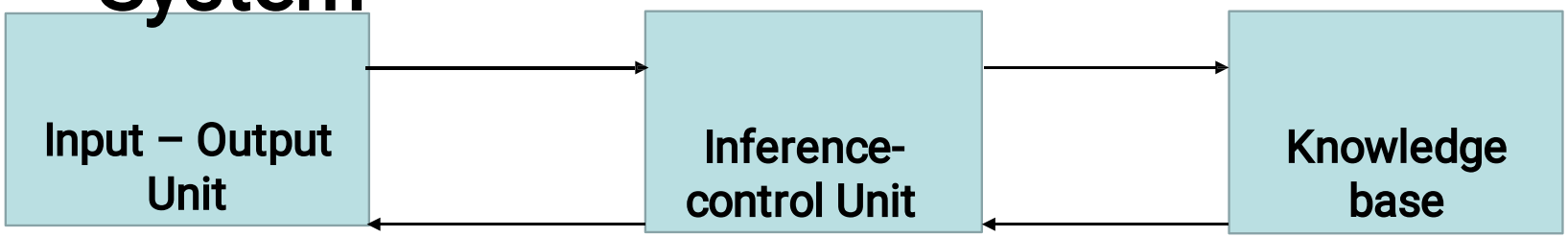
Input-output: Facebook, twitter and movie site comments about the movie.

Knowledge base: Contains the positive, negative and neutral keywords (Dictionary).

Inference-control Unit: AI Algorithm Implementation – Programs

Example: NLP Algorithms, Naïve Bayes Algorithm, Support Vector Machines

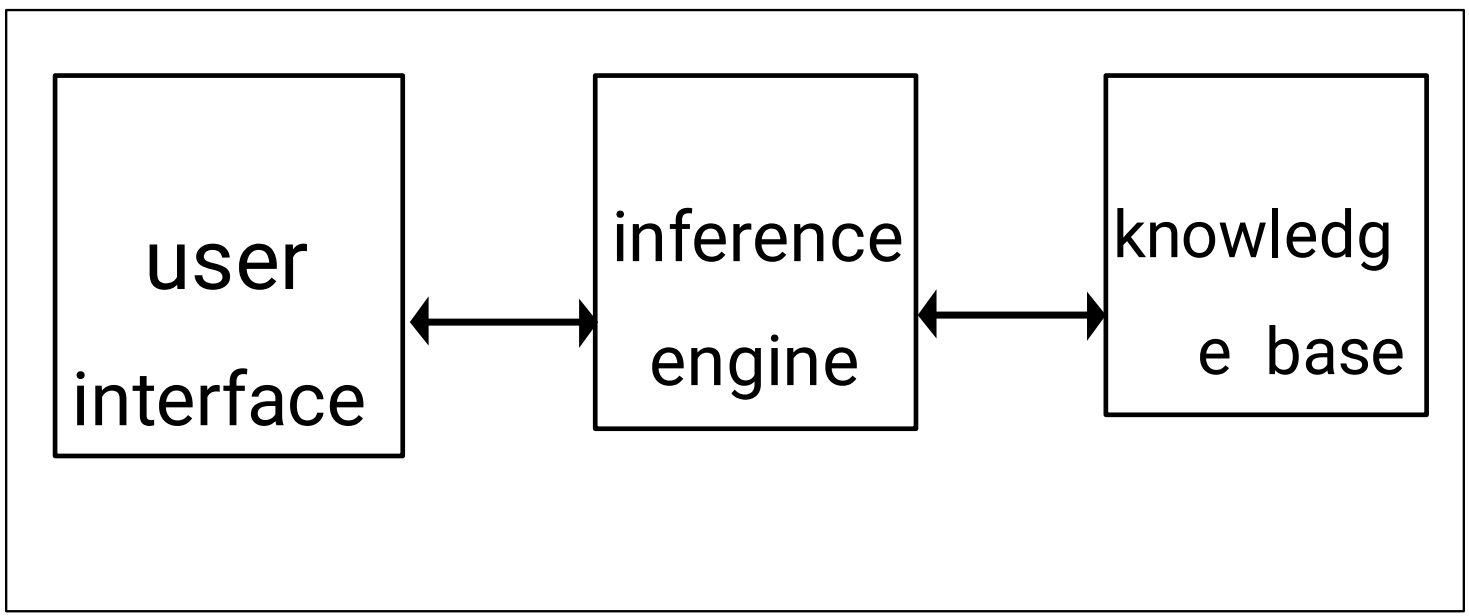
Architecture of AI
Components of Knowledge base System



- Knowledge-based systems – get the power from expert knowledge that has been coded into facts, heuristics, and procedures.
- The knowledge is stored in knowledge base separate from the control and inference components.
- This makes possible to add new knowledge or refine existing knowledge without recompiling the control and inference programs.

KBS architecture (1)

- The typical architecture of an KBS is often described as follows:

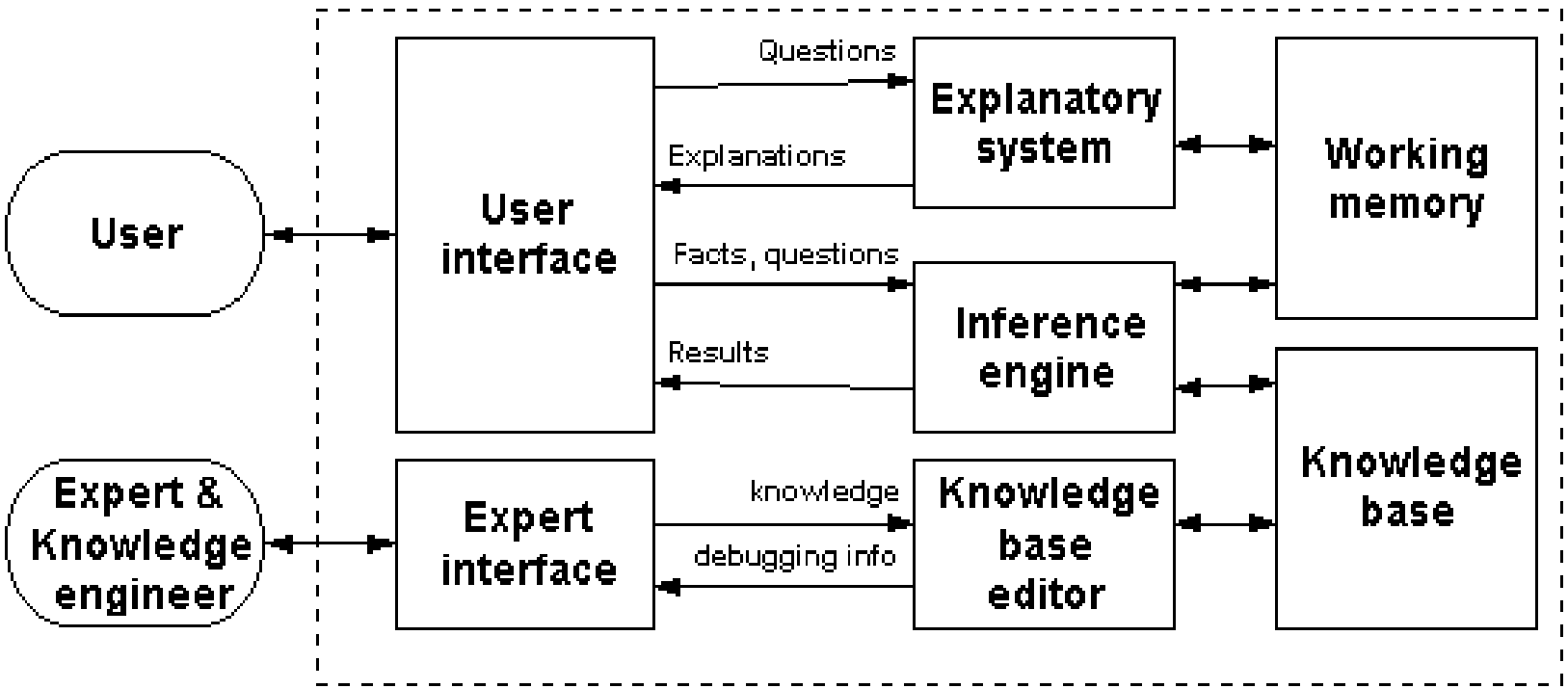


KBS architecture (2)

- However, it is reasonable to produce **a richer, more elaborate, description** of the typical KBS.

- A more **elaborate description**, which still includes the components that are to be found in almost any real-world system, would look like this:

KBS architecture (2)



KBS architecture (2)

- The system holds a collection of ***general principles*** which can potentially be applied to any problem - these are stored in the ***knowledge base*** .

- The system also holds a collection of ***specific details*** that apply to the **current problem** (including **details of how the current reasoning process is progressing**) - these are held in ***working memory*** .

- Both these sorts of information are processed by the ***inference engine*** .

KBS architecture (2)

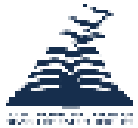
- **Expert System:** This artificial intelligence system is a database which attempts to reason like a person by using **logic** to extract new information from a set of information.

- Any practical expert system needs an **explanatory facility**. It is essential that an expert system should be able to explain its **reasoning**. This is because:

  it gives the user **confidence** in the system;

  it makes it easier to **debug** the system.

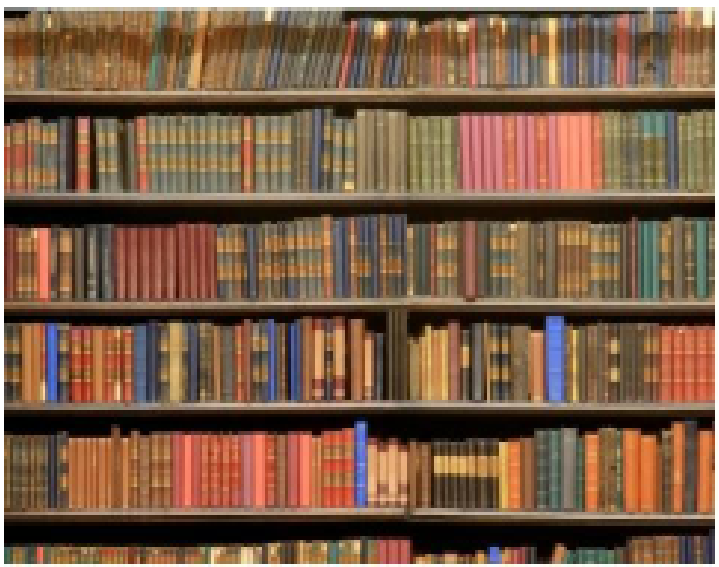
KBS architecture (2)



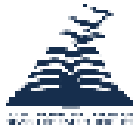
- It is not unreasonable to include an **expert interface & a knowledge base editor**, since any practical KBS is going to need a **mechanism for efficiently building and modifying the knowledge base**.

Knowledge Organization

- Knowledge organizatio involves activitie that *"classify, map, index, and categoriz knowledg for e e navigation, storage, and retrieval "*

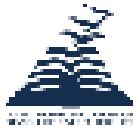


Knowledge Organization



- Organization of knowledge in memory is key to **efficient processing**.
- Knowledge-based systems may require tens of thousands of **facts and rules** to perform their intended tasks.
- It is essential that appropriate facts and rules be easy to **locate and retrieve**.
- Otherwise, much **time will be wasted** in **searching and testing** large number of items in the memory.

Knowledge Organization



- Knowledge can be organized in memory for easy access by a method known as **indexing**.
- It amounts to **grouping the knowledge** in a way that keywords can be used to access the group.

Example:

Positive:	**Negative:**
Nice	Bad
Beautiful	Doubt
Superb	Guilt

Dataset

A collection of related sets of information that is composed of separate elements but can be manipulated as a unit by a computer.

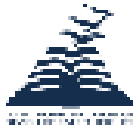
Input File: female.txt

Reshma Akshata
Vani Sita Bhavani
Lalita Ankita Harika

Dataset

- Size of the dataset should be as large as possible, which increases the prediction accuracy.
- Dataset should not contain duplicate elements.

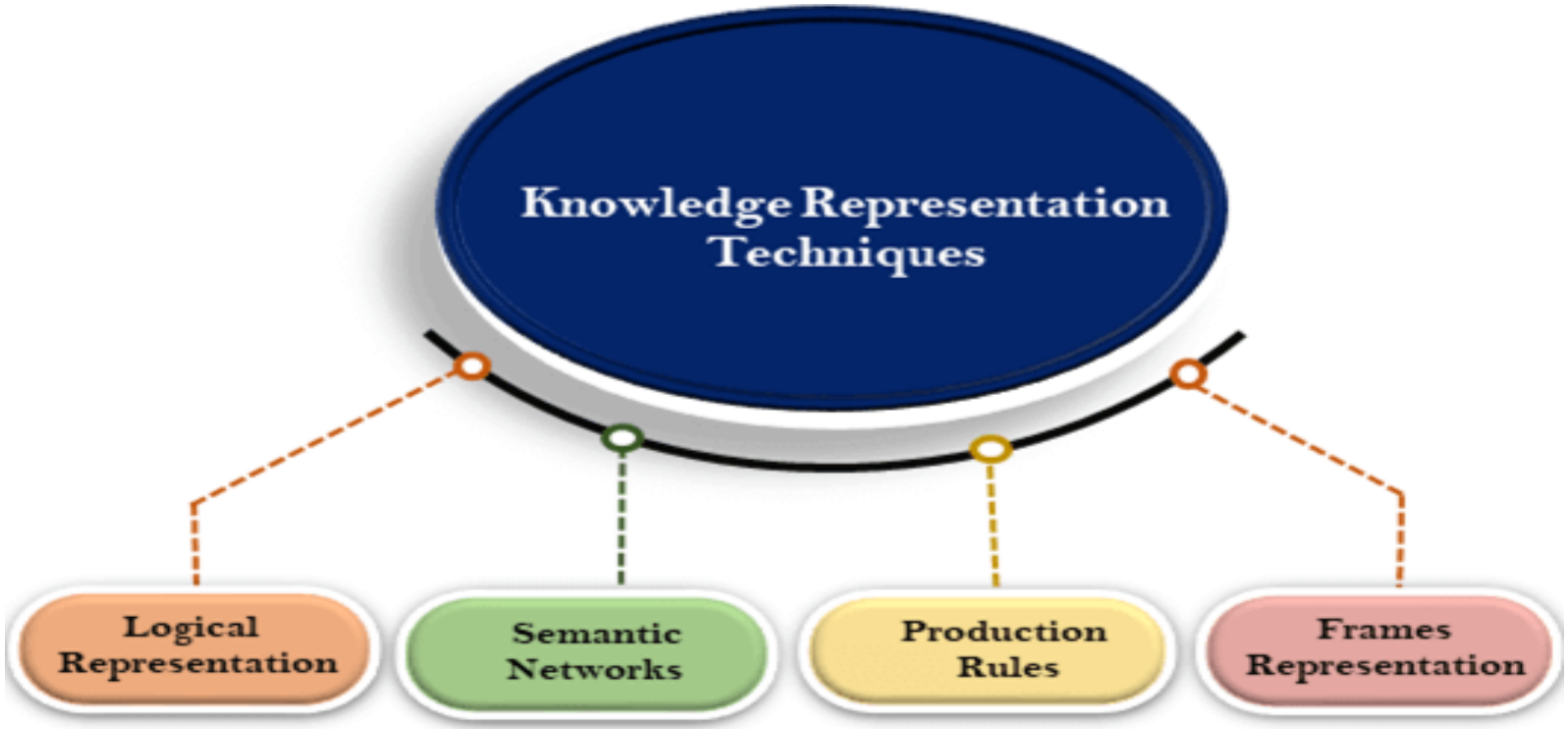
Knowledge Manipulation



- Knowledge manipulation involves the following activities:
  - **Knowledge acquisition**
    - It is the process of gathering, structuring and organizing knowledge of a particular topic or a domain in order to prepare it to be put into the computer memory.
  - **Knowledge Storing**
    - The process of putting the knowledge into the computer in a suitably encoded format.
  - **Knowledge Retrieval**
    - The inverse process of getting the knowledge back whenever it is needed.
  - **Reasoning**
    - This is the most important part of knowledge representation.
    - It includes deriving new knowledge from the existing knowledge by means of an intelligent program.
    - The newly derived knowledge is known as conclusion, inference or explanation.

Knowledge Representation (KR)

- The method used to encode knowledge in an KBS's Knowledge base
- The field of AI dedicated to representing information about the world in a
  form that a computer system can utilize to solve complex tasks



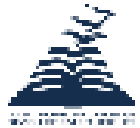
https://www.javatpoint.com/ai-techniques-of-knowledge-representation

Why do we need Knowledge Representation?

- Unlike human mind, computers cannot acquire and represent knowledge by themselves.
- It is complicated to machine process a knowledge represented in natural language.
- Human knowledge is of different types.
- Knowledge manipulation involves:
  - Knowledge acquisition: gathering, structuring and organizing knowledge.
  - Knowledge storing: putting the knowledge into computer.
  - Knowledge retrieval: getting the knowledge when needed.
  - Reasoning: gives conclusion, inference or explanation.

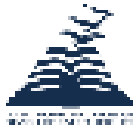
Ref : https://nptel.ac.in/courses/126104006/LectureNotes/Week-3_Knowledge%20Representation.pdf

Knowledge Representation Schemas



• **Logical schemas** - First-order logic - Higher-order logic	• **Network schemas** - Semantic networks - Conceptual graphs
• **Procedural schemas** - Rule-based systems	• **Structural schemas** - Frames - Scripts

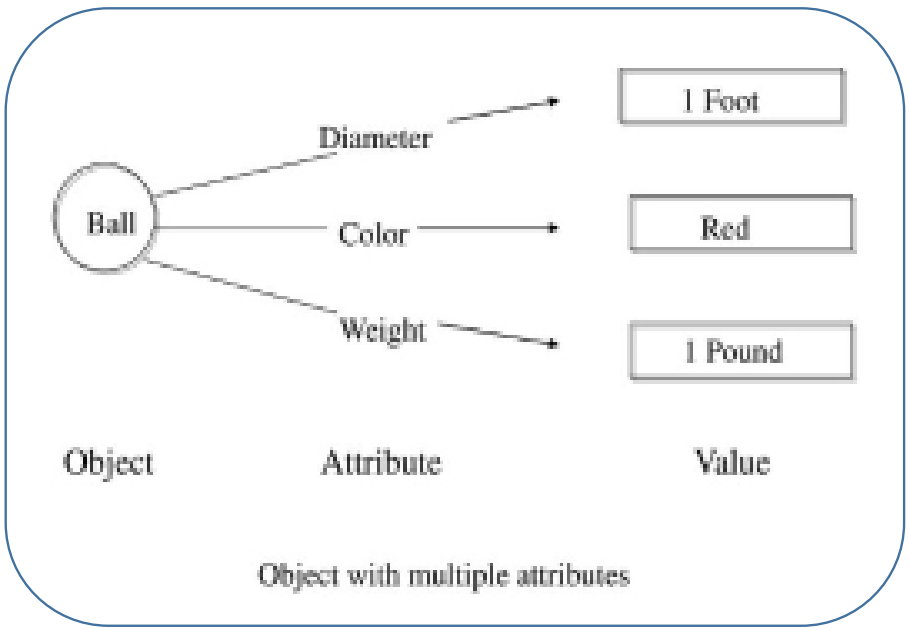
Common Techniques of KR



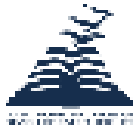
- Object – Attribute – Value Triplets (O-A-V)
- Rules
- Semantic Networks
- Frames
- Logic

O-A-V

- *The Ball is round in shape*
  - **Object** : Ball
  - **Attribute** : Shape
  - **Value** : Red



Rules



- **Rule**: A knowledge structure that relates some known information to other information that can be concluded or inferred
- A rule describes how to solve a problem
- Expert systems employing rules are called rule-based expert systems

Structure of Rule

The rules structure logically connects one or more **antecedents** also called **premises** contained in the **IF** part, to one or more **consequents** also called **conclusions** contained in the **THEN** part.

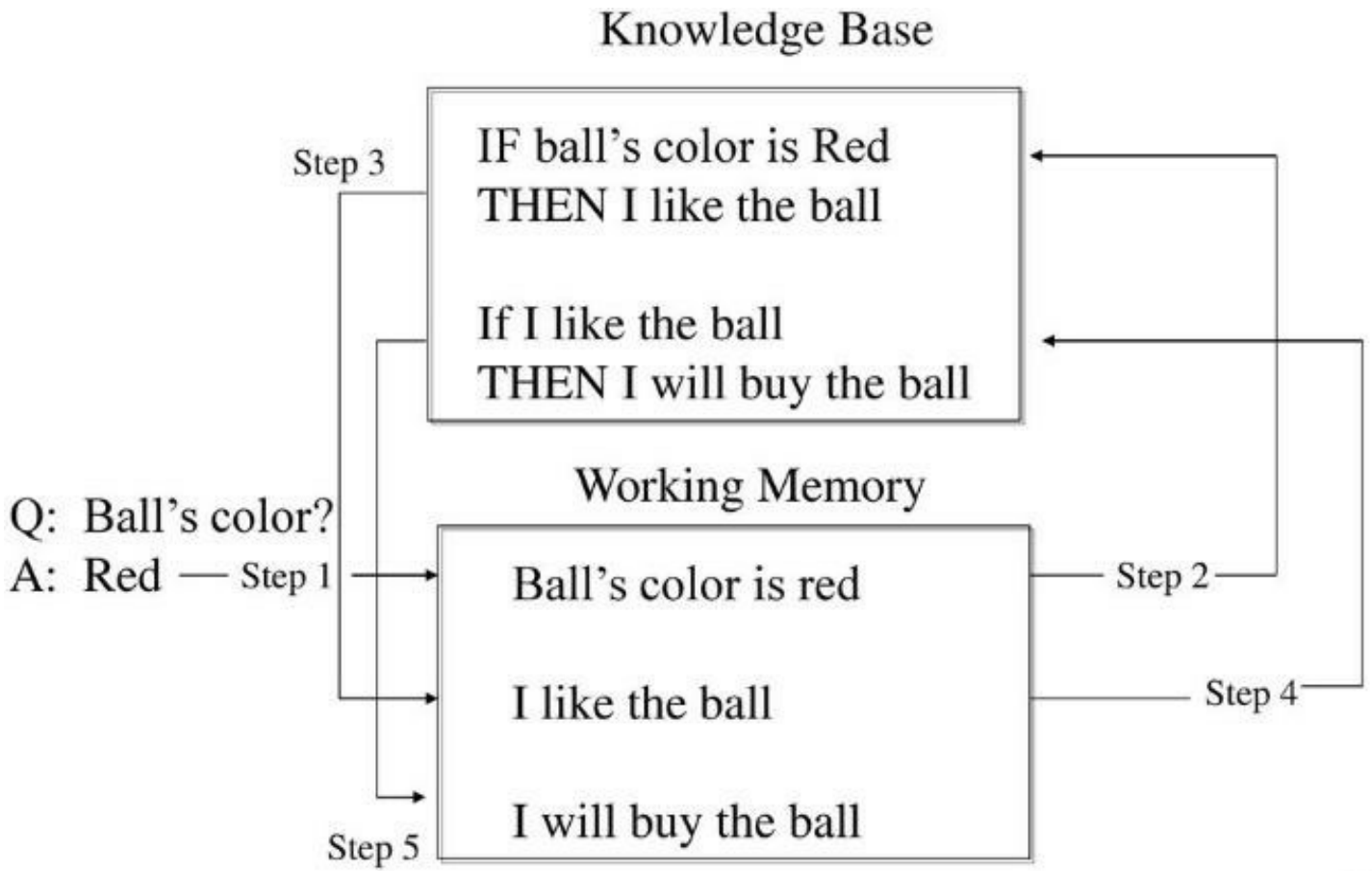
IF	The ball's color is red
THEN	I like the ball

If the balls color is red then we can infer that I like the ball

Working Memory

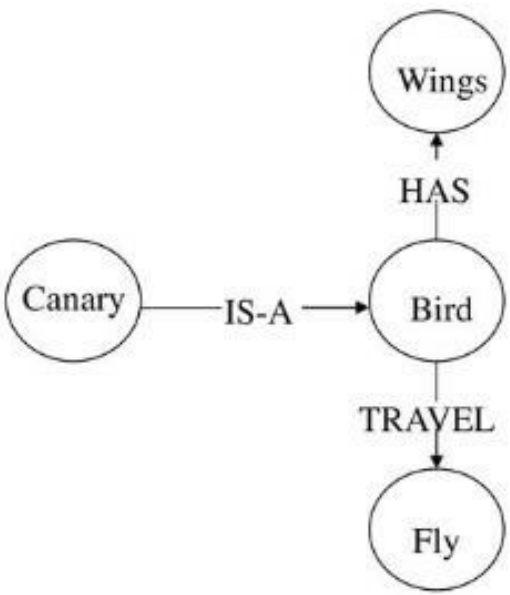
- Expert systems employing rules are called rule-based expert systems
- System matches the IF portion of the rules with facts contained in the working memory
- When a match is confirmed, the rule *fires* and its THEN statements are added to the working memory
- The new statements added to the working memory can also cause other rules to fire

Inference Engine



Semantic Networks

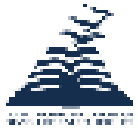
DEFINITION: Semantic Network - A method of knowledge representation using a graph made up of nodes and arcs where the nodes represent objects and the arcs the relationships between the objects



IS-A (Canary, Bird)

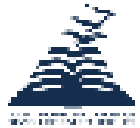
Semantic network of a bird

Associative Networks



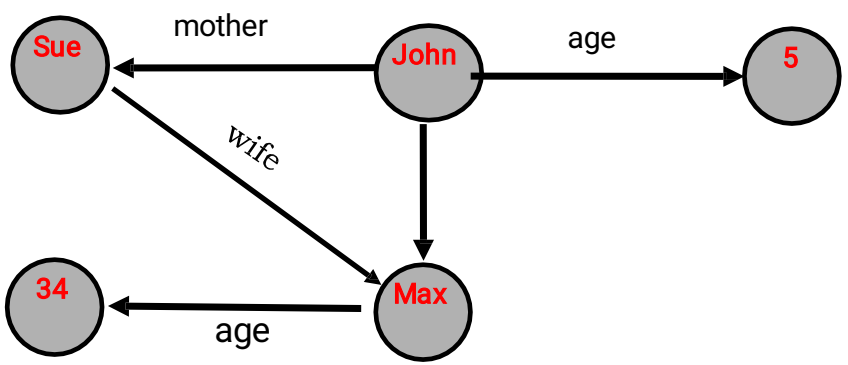
- Semantic networks consist of **nodes, links (edges)** and **link labels**.
- **nodes** appear as circles or ellipses or rectangles to represent objects such as physical objects, concepts or situations.
- **Links** appear as arrows to express the relationships between objects .
- **link labels** specify particular relations .
- As nodes are associated with other nodes semantic nets are also referred to as **Associative Networks.**
- Semantic Networks, Frames and Scripts are sometimes called as **Associative Networks.**

Associative Networks



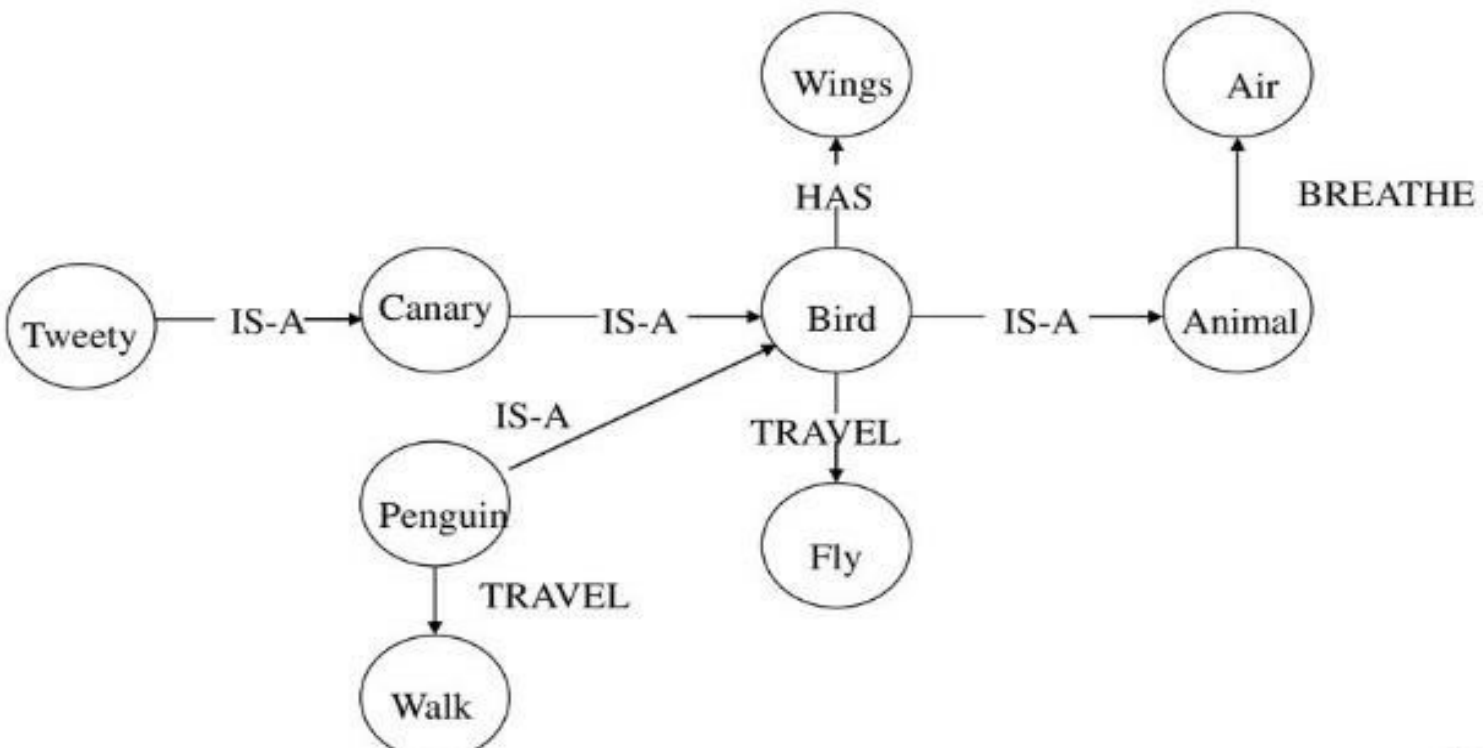
Example :

mother(john,sue)
age(john,5)
wife(sue,max)
age(max,34)



Semantic Networks

You can add a new object node by 1) a similar object, 2) a more specific object, or (3) a more general object.



Basics of Associative Networks

- It's defined as various kinds of links between the **concepts**.

- "has-part" or aggregation.

- "is-a" or specialization.

- More specialized depending on domain.

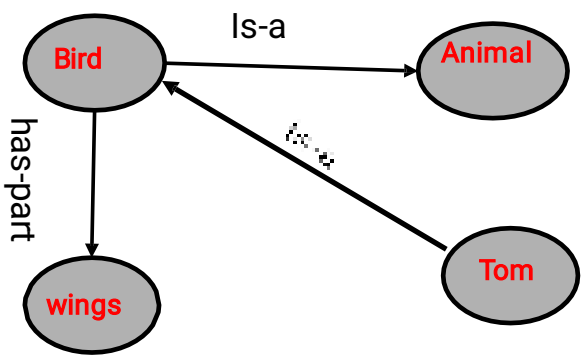
- It typically also includes **Inheritance** and some kind of procedural attachment.

Example :

- Tom is a Bird.
- Bird is a Animal.
- Bird has part Wings.

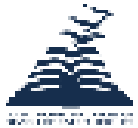
Basics of Associative Networks

Example : Tom is a Bird.
Bird is a Animal.
Bird has part Wings.

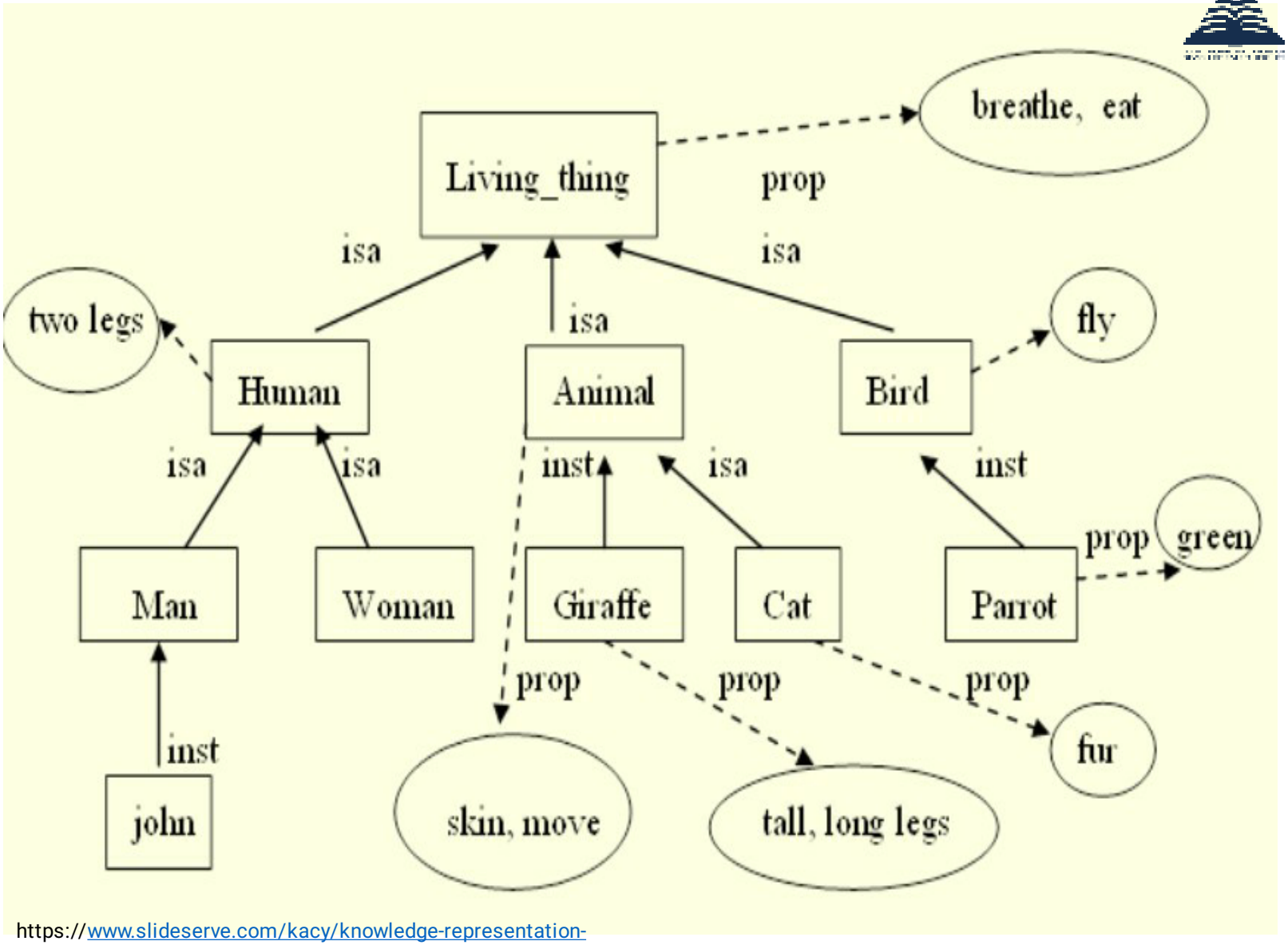


- The ISA (**is-a**) or AKO (a-kind-of) relation is often used to link instances to classes, classes to super classes
- Some links (e.g. **has Part**) are inherited along ISA paths.
- The semantics of a semantic net can be relatively informal or very formal
  - often defined at the implementation level

Draw a Semantic Network

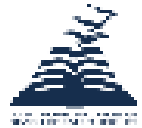


"Every human, animal and bird is living thing who breathe and eat. All birds can fly. All man and woman are humans who have two legs. Cat is an animal and has a fur. All animals have skin and can move. Giraffe is an animal who is tall and has long legs. Parrot is a bird and is green in color".

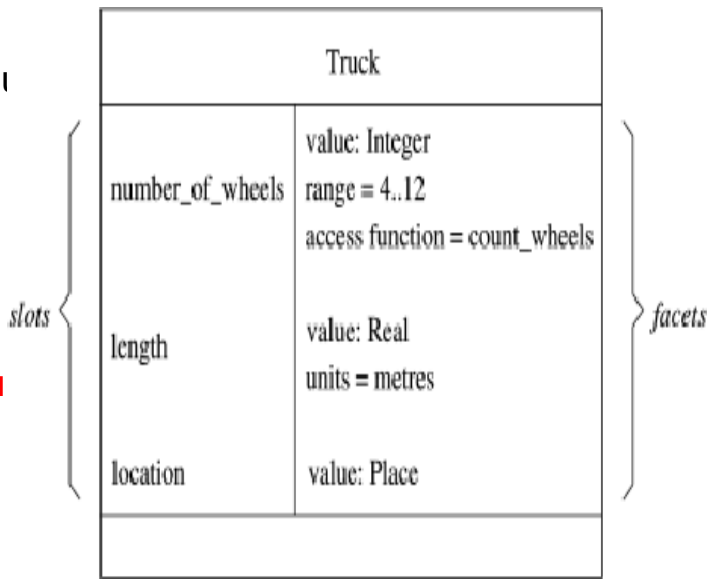


https://www.slideserve.com/kacy/knowledge-representation-techniques

FRAME STRUCTURES

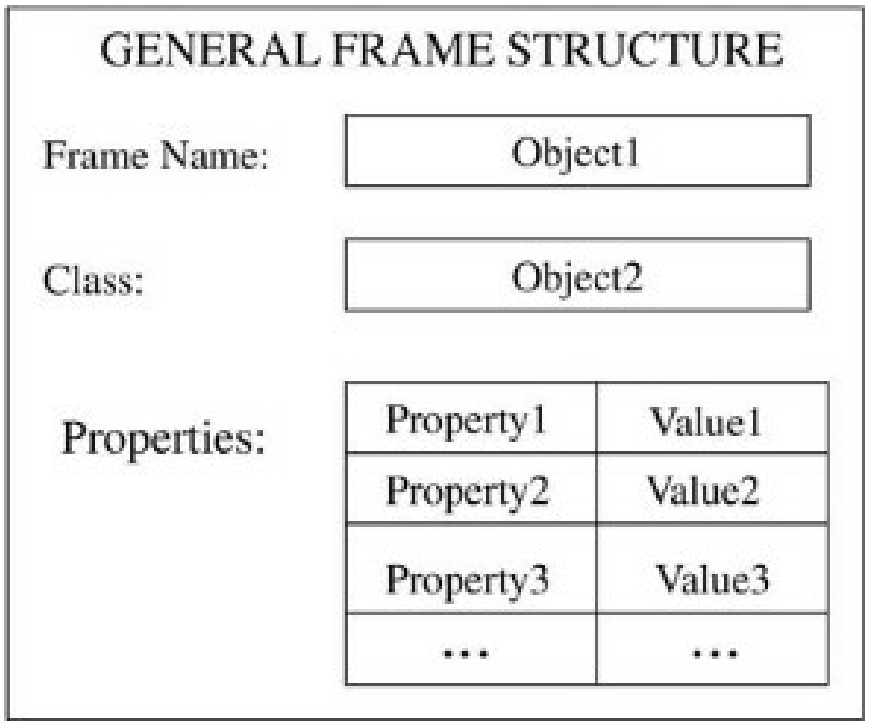


- Semantic networks morphed into Frame representation Languages in the ‘70s and ‘80s.
- A frame is a lot like the notion of an object in OOP, bu has more meta-data.
- Represents related knowledge about a subject
- A frame has a set of slots.
- A slot represents a relation to another frame (or valu
- A slot has one or more facets.
- A facet represents some aspect of the relation.
- Facet : A slot in a frame holds more than a value

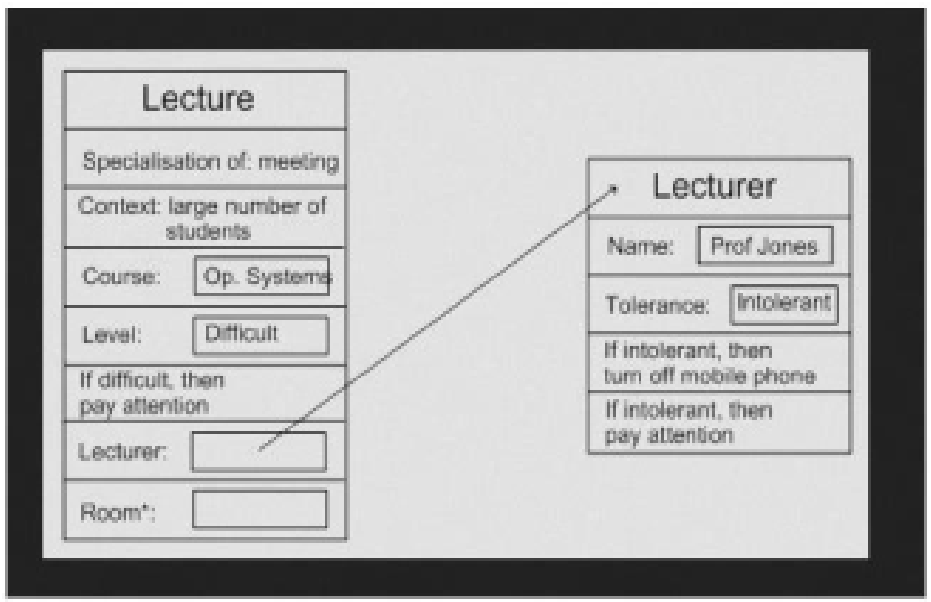


Frame

- A data structure for representing stereotypical knowledge of some concept
- A frame is a collection of attributes and associated values that describe some entity in the world.
- Frames are general record like structures which consist of a collection of slots and slot values.



Frame Structures



Example :

- (Jones)
- (Profession (Value Lecturer))
- (Age (Value 25))
- (City (Value Yelahanka))
- (State (Value Karnataka))

* Note : value is a Keyword.

Class Frame

A class frame represents the general characteristics of some set of common objects.

Frame Name: Bird

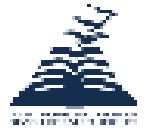
Properties:

Color	Unknown
Eats	Worms
No._Wings	2
Flies	True
Hungry	Unknown
Activity	Unknown

Instance Frame

Instance Frame describes a specific instance of a class frame. The frame inherits both properties and property values from the class.

Frame Name:	Tweety	
Class:	Bird	
Properties:	Color	Yellow
	Eats	Worms
	No._Wings	1
	Flies	False
	Hungry	Unknown
	Activity	Unknown
	Lives	Cage



CLASS FRAME

Frame Name: Bird

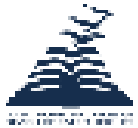
Properties		
	Color	unknown
	Eats	Worms
	#wings	2
	Flies	true
	Hungry	unknown
	Activity	unknown

INSTANCE FRAME

Frame Name: Tweety

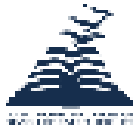
Properties		
	Color	yellow
	Eats	worms
	#wings	1
	Flies	False
	Hungry	unknown
	Activity	unknown

Conceptual Graphs



- Semantic Network where each graph represents a single preposition
- Consists of basic concepts and the relationship between them
- KB consists of set of Conceptual Graphs
- It tries to capture the concepts about the events and represents them in the form of a graph.
- A concept may be individual or generic.
- An individual concept has a type field followed by a reference field.
- A single conceptual graph is roughly equivalent to a graphical diagram of a natural language sentence where the words are depicted as concepts and relationships.

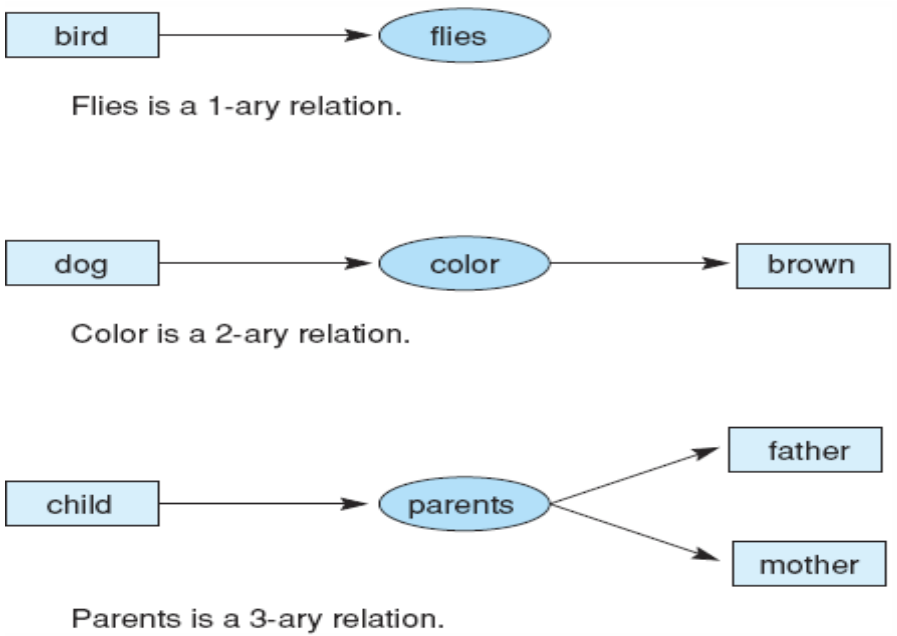
Conceptual Graphs



- **Conceptual graph**
  - A finite, connected, bipartite graph.
  - No arc labels, instead the conceptual relation nodes represent relations between concepts
  - Concepts are represented as boxes and conceptual relations as ellipses
  - Nodes
    - **Concept Nodes – box nodes**
      - Concrete concepts:
        - » These concepts are characterized by our ability to form an image of them in our minds.
        - » cat, telephone, classroom
        - » Concrete concepts include generic concepts such as cat or book along with concepts of specific cats and books
      - Abstract objects:
        - » Abstract Concepts that do not correspond to images in our minds
        - » love, beauty, loyalty
    - **Conceptual Relation Nodes – ellipse nodes**
      - Relations involving one or more concepts
      - Some special relation nodes, namely, agent, recipient, object, experiencer, are used to link a subject and the verb
      - Arity – number of box nodes linked to

Conceptual Graphs

- **Example :**

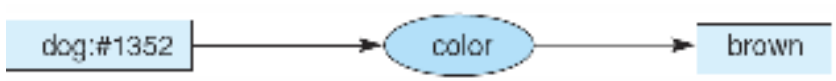


Conceptual Graphs

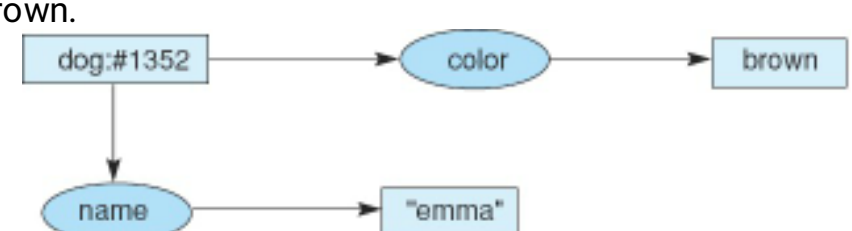
- **Example :**

Conceptual graph indicating that the dog named Emma is brown.

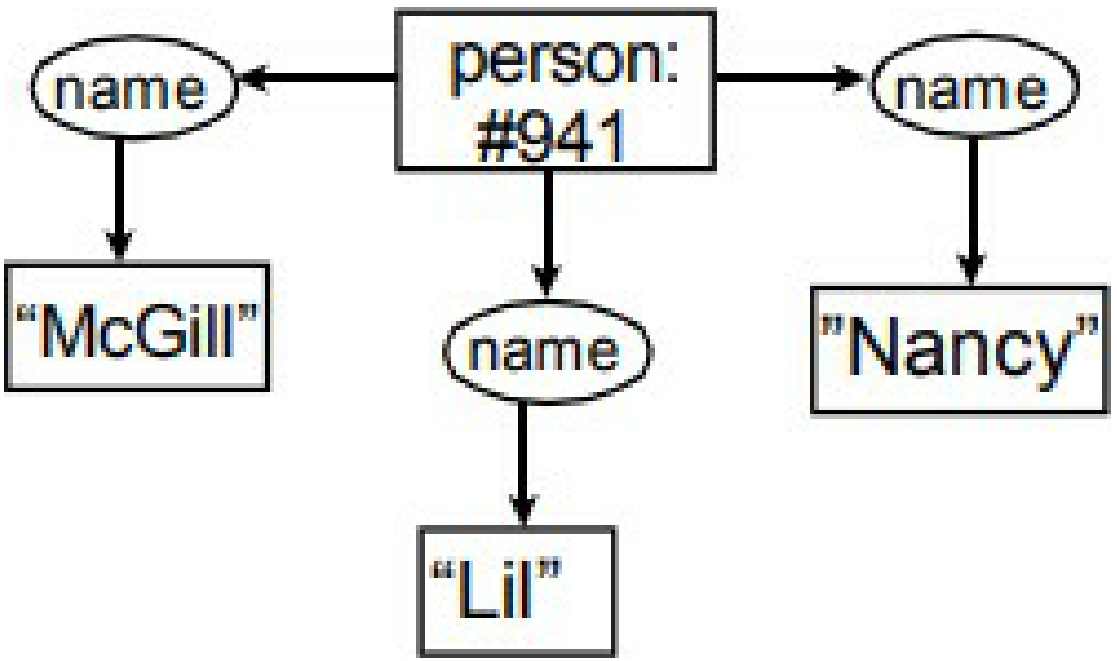
Conceptual graph indicating that a particular (but unnamed) dog is brown.



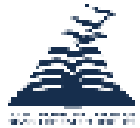
Conceptual graph indicating that a dog named Emma is brown.



Example: Her name was McGill and she called herself Lil, but everyone knew her as Nancy

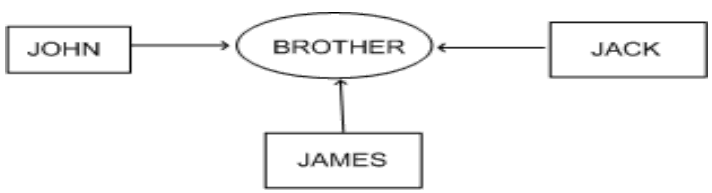


Conceptual Graphs



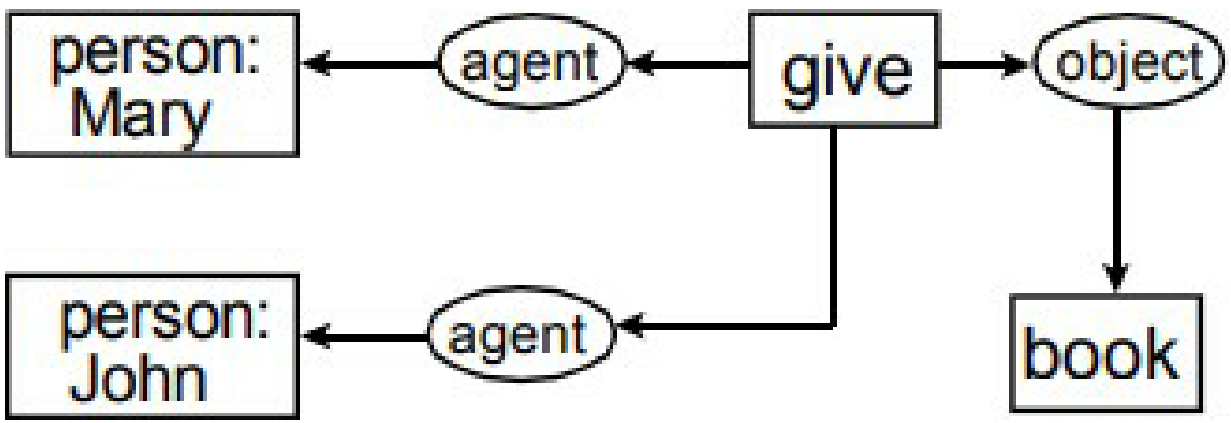
- **Example :**

Each graph represents a single proposition.



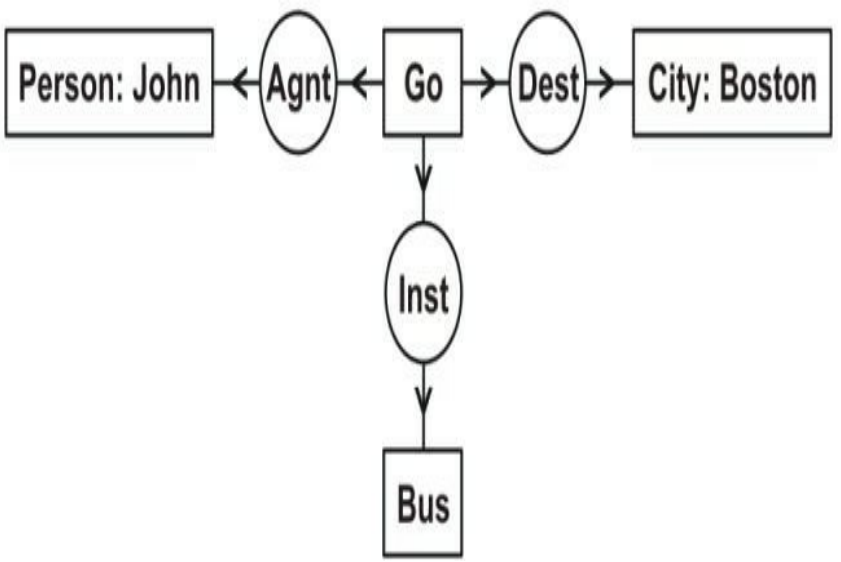
- Advantage:
  - Single relationship between multiple concepts is easily representable.

Example: Mary gave John the book

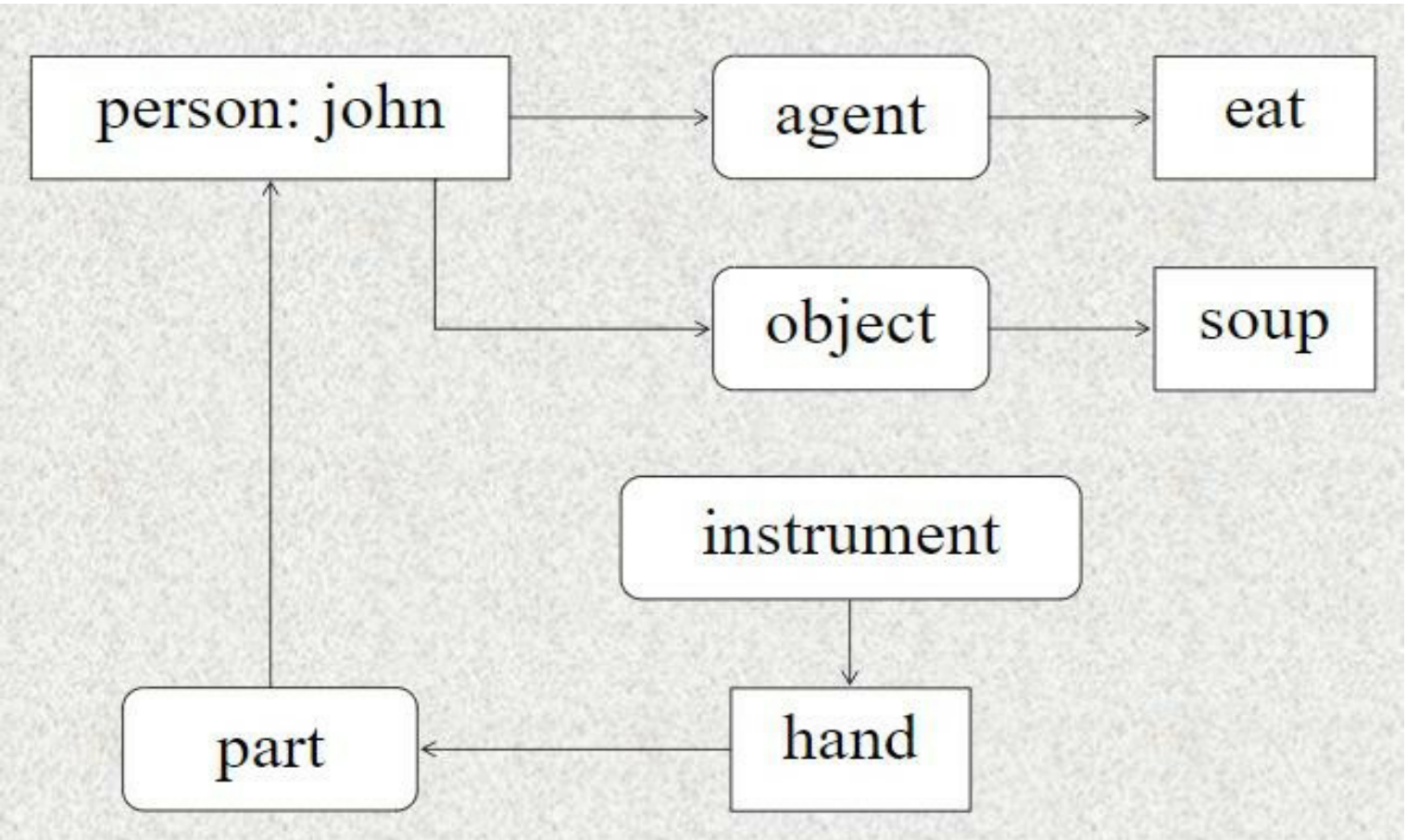


Example: John is going to Boston by bus

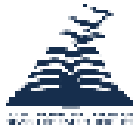
- Each of the four concepts has a type label, which represents the type of entity the concept refers to: **Person, Go, Boston, or Bus**.
- Two of the concepts have names, which identify the referent: John or Boston.
- Each of the three conceptual relations has a type label that represents the type of relation: agent (Agnt), destination (Dest), or instrument (Inst).
- The CG as a whole indicates that the person John is the agent of some instance of going, the city Boston is the destination, and a bus is the instrument.



Example: John agent eat object soup instrument hand part



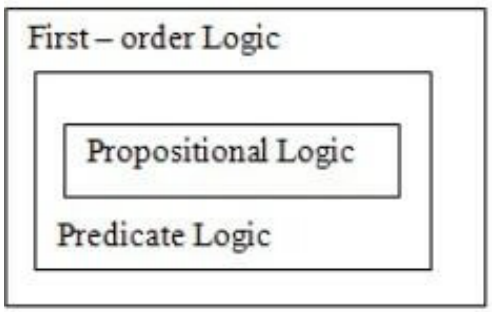
Summary



- Graphs are **very easy to store** inside a computer, by structuring knowledge in a useful fashion
- An **ontology** formally represents concepts in a **domain and relationships**
  - between those concepts
- The concept originated in philosophy; a model of a **theory of nature** or existence.
- An ontology describes the things we want to talk about, **including both**
  - **objects and relationships**

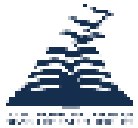
Logic

- Logic is the oldest form of knowledge representation in a computer
- One most often linked to intelligent systems are **propositional logic** and **predicate calculus**
- Techniques use symbols to represent knowledge
- Operators applied to the symbols produce logical reasoning



- A logic is a formal language, with precisely defined syntax and semantics, which supports sound inference.
- Different logics exist, which allow you to represent different kinds of things, and which allow more or less efficient inference.

Acknowledgement & Disclaimer



- The content in the slides is amalgamated from web.
- Many websites have been visited and different tutorials have been used
- I want to thank my colleagues and students for being so supportive and understanding.
- If you find any mistake do point the slide no and point.

Thank you!

Module 2

Logic based Knowledge Representation

Contents

- Introduction
- Propositional Logic
- Predicate Logic
- First order Logic, Properties of well-formed formulas (Wffs)
- Conversion to Clausal Form
- The Resolution Principle, Inference in First Order Logic (FOL)

INTRODUCTION

Introduction

Humans know things; and what they know helps them do things.

How the intelligence of humans is achieved?

by processes of reasoning that operate on internal representations of knowledge.

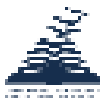
Representation

- AI agents deal with knowledge (data)
  - Facts (believe & observe knowledge)
  - Procedures (how to knowledge)
  - Meaning (relate & define knowledge)
- Right representation is crucial
  - Early realisation in AI
  - Wrong choice can lead to project failure
  - Active research area

In lay man term…

- **Knowledge** -*facts, information, and skills acquired through experience or education; the theoretical or practical understanding of a subject.*
- **Reasoning**-*the action of thinking about something in a logical, sensible way*
  - *The process of thinking about something in order to make a decision*
- **Logic**- *A particular way of thinking, especially one that is reasonable and based on good judgment*
- **Inference-***A guess that you make or an opinion that you form based on the information that you have*
  - Deriving new sentences from old

Knowledge Representation & Reasoning



- AI: The study and development of systems that demonstrate intelligent behavior
- KR&R is the part of AI that is concerned with thinking and how thinking contributes to intelligent behavior
- KR suggests an approach to understanding intelligent behavior that is radically different
- Instead of studying humans very carefully (biology, nervous systems, psychology, sociology, etc.), it argues that what we need to study is ***what humans know*** .
- It is taken as a given that what allows humans to behave intelligently is that they know a lot of things about a lot of things and are able to apply this knowledge as appropriate to adapt to their environment and achieve their goals.
- KR&R focuses on the knowledge, not on the knower. We ask what *any* agent—human, animal, electronic, mechanical—would need to know to behave intelligently, and what sort of computational mechanisms might allow its knowledge to be manipulated.

Knowledge Based Agents

- In AI, this approach to intelligence is embodied in **knowledge-based agents** - composed of a knowledge base and an inference mechanism.

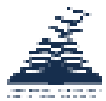
It operates by

- storing sentences about the world in its knowledge base,
- using the inference mechanism to infer new sentences,
- using these sentences to decide what action to take.

Knowledge Based Agents..

- The central component of a knowledge-based agent is its **knowledge base**, or KB.
- A knowledge base is a set of **sentences**.
- Each sentence is expressed in a language called a **knowledge representation language.**
- There must be a way to add new sentences to the knowledge base (TELL)and a way to query what is known(ASK).
- The standard names for these operations are TELL and ASK , respectively.

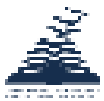
Knowledge



- “Amith knows that sun will rise tomorrow
- Knowledge is a relation between a **knower** and a **proposition**
  - **knower :** Amith
  - **proposition:** the idea expressed by a simple declarative sentence, like “sun will rise tomorrow”

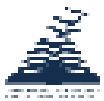
- For KR&R, what matters about propositions is that they are abstract entities that can be true or false, right or wrong
  - When we say, “Amith knows that *p* ,” we can just as well say, “Amith knows that it is true that *p* .”

Reasoning

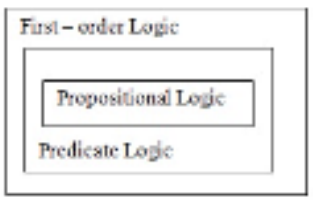


- What is **reasoning**? In general, it is the formal manipulation of the symbols representing a collection of believed propositions to produce representations of new ones.
  - Symbols are more accessible than the propositions they represent: They must be concrete enough that we can manipulate them (move them around, take them apart, copy them, string them together) in such a way as to construct representations of new propositions.
    - We might start with the sentences *"Amith loves Buzo"* and *"Buzo is hurt"* and after a certain amount of manipulation produce the sentence, *"Someone Amith loves is hurt"*
    - We would call this form of reasoning ***logical inference*** because the final sentence represents a logical conclusion of the propositions represented by the initial ones
- Reasoning is a form of calculation, not unlike arithmetic, but over symbols standing for propositions rather than numbers

LOGIC



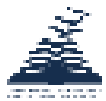
- A logic is a formal language, with precisely defined syntax and semantics, which supports sound inference.
- Different logics exist, which allow you to represent different kinds of things, and which allow more or less efficient inference.
- The logic may be different types like propositional logic, predicate logic, temporal logic, description logic etc.



What is a Logic?

- A language with concrete rules
  - No ambiguity in representation (may be other errors!)
  - Allows unambiguous communication and processing
  - Very unlike natural languages e.g. English
- Many ways to translate between languages
  - A statement can be represented in different logics
  - And perhaps differently in same logic
- **Expressiveness** of a logic
  - How much can we say in this language?
- Not to be confused with logical reasoning
  - Logics are languages, reasoning is a process (may **use** logic)

Logic Representation



- Logic is a great knowledge representation language for many AI problems
- **Propositional logic** is the simple foundation and fine for some AI problems
- **First order logic** (FOL) is much more expressive as a KR language and more commonly used in AI
- There are many variations: horn logic, higher order logic, three-valued logic, probabilistic logics, etc.

Syntax and Semantics

- Syntax
  - Rules for constructing legal sentences in the logic
  - Which symbols we can use (English: letters, punctuation)
  - How we are allowed to combine symbols
- Semantics
  - How we interpret (read) sentences in the logic
  - Assigns a meaning to each sentence
- Example: “All lecturers are seven foot tall”
  - A valid sentence (syntax)
  - And we can understand the meaning (semantics)
  - This sentence happens to be false (there is a counterexample)

Propositional Logic

- Syntax
  - Propositions, e.g. “it is wet”
  - Connectives: and, or, not, implies, iff (equivalent)

    $\wedge \quad \vee \quad \neg \quad \rightarrow \quad \leftrightarrow$
  - Brackets, T (true) and ⊢ (false)
- Semantics (Classical AKA Boolean)
  - Define how connectives affect truth
    - “P and Q” is true if and only if P is true and Q is true
  - Use **truth tables** to work out the truth of statements

Predicate Logic

- Propositional logic combines atoms
  - An atom contains no propositional connectives
  - Have no structure (today_is_wet, john_likes_apples)
- **Predicates** allow us to talk about objects
  - Properties: is_wet(today)
  - Relations: likes(john, apples)
  - True or false
- In predicate logic each atom is a predicate
  - e.g. first order logic, higher-order logic

First Order Logic

- More expressive logic than propositional
  - Used in this course (Lecture 6 on representation in FOL)
- **Constants** are objects: john, apples
- **Predicates** are properties and relations:
  - likes(john, apples)
- **Functions** transform objects:
  - likes(john, fruit_of(apple_tree))
- **Variables** represent any object: likes(X, apples)
- **Quantifiers** qualify values of variables
  - True for all objects (Universal): $\forall$X. likes(X, apples)
  - Exists at least one object (Existential): $\exists$X. likes(X, apples)

Example: FOL Sentence

- “Every rose has a thorn”

$$\forall X.\big(rose(X) \rightarrow \exists Y.(has(X,Y) \wedge thorn(Y))\big)$$

- For all X
  - if (X is a rose)
  - then there exists Y
    - (X has Y) and (Y is a thorn)

Propositional Logic

PROPOSITIONAL LOGIC

- A **proposition** is a declarative sentence (a sentence that declares a fact) that is either true or false, but not both.
- Are the following sentences propositions?
  - Toronto is the capital of Canada. (Yes)
  - Read this carefully. (No)
  - 1+2=3 (Yes)
  - x+1=2 (No)
  - What time is it? (No)

Continued..

- Propositional logic (PL) is the simplest form of logic where all the statements are made by propositions.
- A proposition is a declarative statement which is either true or false.
- **It is a technique of knowledge representation in logical and mathematical form.**

Examples:

- Today is Sunday.
- The Sun rises from West (False proposition)
- 3+3= 7 (False proposition)
- 5 is a prime number
- Tomorrow is Holiday.

Continued..

- Propositional Logic – the area of logic that deals with propositions
- Propositional Variables – variables that represent propositions: p q r s
  - E.g. Proposition p – “Today is Friday.”
- Truth values – T, F

Syntax

- The syntax of propositional logic defines the allowable sentences.
- The **atomic sentences** consist of a **single proposition symbol**.
- We use symbols that start with an uppercase letter and may contain other letters or subscripts
- There are two proposition symbols with fixed meanings: **True** is the always-true proposition and **False** is the always-false proposition.
- **Complex sentences** are constructed from simpler sentences, using **parentheses and logical connectives**.

Example:

- 2+2 is 4, it is an atomic proposition as it is a true fact.
- "The Sun is cold" is also a proposition as it is a false fact.
- Compound proposition - Compound propositions are constructed by combining simpler or atomic propositions, using parenthesis and logical connectives.

Example:

- "It is raining today, and street is wet."
- "Ankit is a doctor, and his clinic is in Mumbai."

Semantics

The semantics defines the rules for determining the truth of a sentence with respect to a particular model.

In propositional logic, a model simply fixes the truth value—true or false—for every proposition symbol.

- For example, if the sentences in the knowledge base make use of the proposition symbols P1,2, P2,2, and P3,1, then one possible model is

  m1 = {P1,2 =false, P2,2 =false, P3,1 =true} .

- With three proposition symbols, there are 23 =8 possible models

1.1 Propositional Logic

DEFINITION 1

Let p be a proposition. The negation of p, denoted by $\neg p$, is the statement "It is not the case that p."

The proposition $\neg p$ is read "not p." The truth value of the negation of p, $\neg p$ is the opposite of the truth value of p.

- Examples
  - Find the negation of the proposition "Today is Friday." and express this in simple English.

    Solution: The negation is "It is not the case that *today is Friday*."

    In simple English, "Today is not Friday." or "It is not Friday today."
  - Find the negation of the proposition "At least 10 inches of rain fell today in Miami." and express this in simple English.

    Solution: The negation is "It is not the case that *at least 10 inches of rain fell today in Miami*."

    In simple English, "Less than 10 inches of rain fell today in Miami."

1.1 Propositional Logic

- Note: Always assume fixed times, fixed places, and particular people unless otherwise noted.
- Truth table:

The Truth Table for the Negation of a Proposition.	
p	$\neg p$
T	F
F	T

- Logical operators are used to form new propositions from two or more existing propositions. The logical operators are also called connectives.

Logical connectives.

Symbol	Meaning	priority
¬	negation, no	1
∧	conjunction, and	2
∨	disjunction, or	3
⇒	implication	4
⇔	equivalence	5

¬(not) -- ¬P is called the **negation**. Either an atomic sentence (a **positive literal**) or a negated atomic sentence (a **negative literal**)
.

∧ (and) -- P ∧ Q, is called a **conjunction**; its parts are the **conjuncts**.

[P= Rohan is intelligent, Q= Rohan is hardworking - P∧ Q]

∨ (or) -- P∨Q, is a **disjunction** of the **disjuncts** P and Q

[P= Ritika is Doctor. Q= Ritika is Engineer - P ∨ Q]

Connective symbols	Word	Technical term	Example
∧	AND	Conjunction	A ∧ B
∨	OR	Disjunction	A ∨ B
→	Implies	Implication	A → B
⇔	If and only if	Biconditional	A ⇔ B
¬ or ~	Not	Negation	¬ A or ¬ B

Logical connectives.

⇒ (implies) -- P ⇒ ¬Q, is an **implication** (or **conditional**). Its **premise** or **antecedent** is **P,** and its **conclusion** or **consequent** is ¬**Q**. Implications are also known as **rules** or **if–then** statements. The implication symbol is sometimes written as ⊃ or → .

If it is raining, then the street is wet.

[P= It is raining Q= Street is wet - P → Q]

⇔ (**equivalence)** -- P ⇔ ¬Q is a **biconditional**. **If and only if** statements. Some books write this as ≡

If I am breathing, then I am alive

Truth tables for the five logical connectives

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

Propositional Logic

DEFINITION 2

Let p and q be propositions. The *conjunction* of p and q, denoted by $p \wedge q$, is the proposition "p and q". The conjunction $p \wedge q$ is true when both p and q are true and is false otherwise.

- Examples
  - Find the conjunction of the propositions p and q where p is the proposition "Today is Friday." and q is the proposition "It is raining today.", and the truth value of the conjunction.

    Solution: The conjunction is the proposition "Today is Friday and it is raining today." The proposition is true on rainy Fridays.

Propositional Logic

DEFINITION 3

Let p and q be propositions. The *disjunction* of p and q, denoted by $p \vee q$, is the proposition "p or q". The conjunction $p \vee q$ is false when both p and q are false and is true otherwise.

- Note:

  inclusive or: The disjunction is true when at least one of the two propositions is true.

  - E.g. "Students who have taken calculus or computer science can take this class." – those who take one or both classes.

  exclusive or: The disjunction is true only when one of the proposition is true.

  - E.g. "Students who have taken calculus or computer science, but not both, can take this class." – only those who take one of them.

- Definition 3 uses *inclusive or*.

Propositional Logic

DEFINITION 4

Let p and q be propositions. The *exclusive or* of p and q, denoted by $p \oplus q$, is the proposition that is true when exactly one of p and q is true and is false otherwise.

The Truth Table for the Conjunction of Two Propositions.

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

The Truth Table for the Disjunction of Two Propositions.

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

The Truth Table for the Exclusive *Or* (*XOR*) of Two Propositions.

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

Propositional Logic

Conditional Statements

DEFINITION 5

Let p and q be propositions. The *conditional statement* $p \rightarrow q$, is the proposition "if p, then q." The conditional statement is false when p is true and q is false, and true otherwise. In the conditional statement $p \rightarrow q$, p is called the *hypothesis* (or *antecedent* or *premise*) and q is called the *conclusion* (or *consequence*).

- A conditional statement is also called an implication.
- Example: "If I am elected, then I will lower taxes." $p \rightarrow q$

implication:

elected, lower taxes.	T T | T
not elected, lower taxes.	F T | T
not elected, not lower taxes.	F F | T
elected, not lower taxes.	T F | F

Propositional Logic

- Example:
  - Let p be the statement "Maria learns discrete mathematics." and q the statement "Maria will find a good job." Express the statement $p \rightarrow q$ as a statement in English.

    Solution: Any of the following -

    "If Maria learns discrete mathematics, then she will find a good job.

    "Maria will find a good job when she learns discrete mathematics."

    "For Maria to get a good job, it is sufficient for her to learn discrete mathematics."

    "Maria will find a good job unless she does not learn discrete mathematics."

Propositional Logic

DEFINITION 6

Let p and q be propositions. The *biconditional statement* $p \leftrightarrow q$ is the proposition "p if and only if q." The biconditional statement $p \leftrightarrow q$ is true when p and q have the same truth values, and is false otherwise. Biconditional statements are also called *bi-implications*.

- $p \leftrightarrow q$ has the same truth value as $(p \rightarrow q) \wedge (q \rightarrow p)$
- *"if and only if"* can be expressed by *"iff"*
- Example:
  - Let p be the statement "You can take the flight" and let q be the statement "You buy a ticket." Then $p \leftrightarrow q$ is the statement

    "You can take the flight if and only if you buy a ticket."

    Implication:

    If you buy a ticket you can take the flight.

    If you don't buy a ticket you cannot take the flight.

Propositional Logic

The Truth Table for the Biconditional $p \leftrightarrow q$.		
p	q	$p \leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

Propositional Logic

Truth Tables of Compound Propositions

- We can use connectives to build up complicated compound propositions involving any number of propositional variables, then use truth tables to determine the truth value of these compound propositions.
- Example: Construct the truth table of the compound proposition

  $(p \vee \neg q) \rightarrow (p \wedge q)$.

The Truth Table of $(p \vee \neg q) \rightarrow (p \wedge q)$.

p	q	$\neg q$	$p \vee \neg q$	$p \wedge q$	$(p \vee \neg q) \rightarrow (p \wedge q)$
T	T	F	T	T	T
T	F	T	T	F	F
F	T	F	F	F	T
F	F	T	T	F	F

Propositional Logic

Precedence of Logical Operators

- We can use parentheses to specify the order in which logical operators in a compound proposition are to be applied.
- To reduce the number of parentheses, the precedence order is defined for logical operators.

Precedence of Logical Operators.	
Operator	Precedence
¬	1
∧	2
∨	3
→	4
↔	5

E.g. $\neg p \wedge q = (\neg p) \wedge q$

$p \wedge q \vee r = (p \wedge q) \vee r$

$p \vee q \wedge r = p \vee (q \wedge r)$

Propositional Logic

Translating English Sentences

- English (and every other human language) is often ambiguous. Translating sentences into compound statements removes the ambiguity.
- Example: How can this English sentence be translated into a logical expression?

"You cannot ride the roller coaster if you are under 4 feet tall unless you are older than 16 years old."

Solution: Let q, r, and s represent "You can ride the roller coaster," "You are under 4 feet tall," and "You are older than 16 years old." The sentence can be translated into:

$$(r \wedge \neg s) \rightarrow \neg q.$$

Propositional Logic

- Example: How can this English sentence be translated into a logical expression?

  "You can access the Internet from campus only if you are a computer science major or you are not a freshman."

  Solution: Let a, c, and f represent "You can access the Internet from campus," "You are a computer science major," and "You are a freshman." The sentence can be translated into:

  $$a \rightarrow (c \vee \neg f).$$

Interpretation of logic formulas

- When we see a logic formula, we do not care about the original meaning of each proposition. Rather, we care about the logic relation between the propositions.
- The main concern is to know if a logic formula is "**true**" or "**false**" when the values of the atomic formulas contained in this formula are given.
- The process for determining the "true/false" of a logic formula based on the "true/false" of the atomic formulas is called "**interpretation**".

Wff- well formed formula

- A well-formed formula, wff, is a finite sequence of symbols from a given alphabet that is part of a formal language. A formal language can be identified with the set of formulas in the language. We represent propositions by formulas called well-formed formulas (wffs).
- A sentence (wff- well formed formula) is defined as follows:
  - A symbol is a sentence
  - If S is a sentence, then $\neg$S is a sentence
  - If S is a sentence, then (S) is a sentence
  - If S and T are sentences, then (S $\vee$ T), (S $\wedge$ T), (S $\rightarrow$ T), and (S $\leftrightarrow$ T) are sentences
  - A sentence results from a finite number of applications of the rules

Properties of wffs in PL

Types of logic formula

Tautology: A tautology is a logic formula that is always true regardless of the true/false of the atomic formulas. A tautology is always "**valid**".

Contradiction : If a logic formula is always false regardless of the true/false of the atomic formulas, we say it is a contradiction. A contradiction is **unsatisfiable**.

Satisfiable: If there exists a set of values for the atomic formulas that makes the logic formula true, this formula is satisfiable. This set of values is called an **interpretation** of the formula.

Equivalence- The wff V is *equivalent* to the wff W (written $V \equiv W$) iff V and W have the same truth value for each assignment of truth values to the propositional variables occurring in V and W.

- *Example* . $\neg A \wedge (B \vee A) \equiv \leftarrow A \wedge B$ and $A \vee \neg A \equiv B \vee \neg B$

Properties of wffs in PL

- Propositional logic is also called Boolean logic as it works on 0 and 1.
- Propositions can be either true or false, but it cannot be both.
- Statements which are questions, commands, or opinions are not propositions such as "Where is Rohini", "How are you", "What is your name", are not propositions.

Properties of wffs in PL- Logical equivalence

- Logical equivalence is one of the features of propositional logic. Two propositions are said to be logically equivalent if and only if the columns in the truth table are identical to each other.
- Let's take two propositions A and B, so for logical equivalence, we can write it as $A \Leftrightarrow B$. In below truth table we can see that column for $\neg A \vee B$ and $A \rightarrow B$, are identical hence A is Equivalent to B

A	B	¬A	¬A V B	A→B
T	T	F	**T**	**T**
T	F	F	**F**	**F**
F	T	T	**T**	**T**
F	F	T	**T**	**T**

Properties of wffs in PL - Properties of Operators:

Commutativity:

P∧ Q= Q ∧ P, or

P ∨ Q = Q ∨ P.

Associativity:

(P ∧ Q) ∧ R= P ∧ (Q ∧ R),

(P ∨ Q) ∨ R= P ∨ (Q ∨ R)

Identity element:

P ∧ True = P,

P ∨ True= True.

Distributive:

P∧ (Q ∨ R) = (P ∧ Q) ∨ (P ∧ R).

P ∨ (Q ∧ R) = (P ∨ Q) ∧ (P ∨ R).

DE Morgan's Law:

¬ (P ∧ Q) = (¬P) ∨ (¬Q)

¬ (P ∨ Q) = (¬ P) ∧ (¬Q)

Double-negation elimination:

¬ (¬P) = P.

$$
\begin{aligned}
(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) && \text{commutativity of } \wedge \\
(\alpha \vee \beta) &\equiv (\beta \vee \alpha) && \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) && \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) && \text{associativity of } \vee \\
\neg(\neg\alpha) &\equiv \alpha && \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) && \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \vee \beta) && \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) &\equiv (\neg\alpha \vee \neg\beta) && \text{De Morgan} \\
\neg(\alpha \vee \beta) &\equiv (\neg\alpha \wedge \neg\beta) && \text{De Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) && \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) && \text{distributivity of } \vee \text{ over } \wedge
\end{aligned}
$$

Standard logical equivalences. The symbols α, β, and γ stand for arbitrary sentences of propositional logic.

Example

1. It is not sunny this afternoon and it is colder than yesterday.
2. If we go swimming it is sunny.
3. If we do not go swimming then we will take a canoe trip.
4. If we take a canoe trip then we will be home by sunset.
5. We will be home by sunset

p It is sunny this afternoon

q It is colder tha n yesterday

r We go swimming

s We will take a canoe trip

t We will be home by sunset (the conclusion)

propositions

1. $\neg p \wedge q$
2. $r \rightarrow p$
3. $\neg r \rightarrow s$
4. $s \rightarrow t$
5. t

hypotheses

Inference rules

- Logical inference creates new sentences that logically follow from a set of sentences (KB)
- An inference rule is **sound** if every sentence X it produces when operating on a KB logically follows from the KB
  - i.e., inference rule creates no contradictions
- An inference rule is **complete** if it can produce every expression that logically follows from (is entailed by) the KB.
- **Logical inference** is used to create new sentences that logically follow from a given set of sentences (KB).
- In artificial intelligence, we need intelligent computers which can create new logic from old logic or by evidence, so generating the conclusions from evidence and facts is termed as Inference.

Rules of Inference

- Inference rules are the templates for generating valid arguments. Inference rules are applied to derive proofs in artificial intelligence, and the proof is a sequence of the conclusion that leads to the desired goal.
- Following are some terminologies related to inference rules:
- Implication: It is one of the logical connectives which can be represented as P → Q. It is a Boolean expression.
- Converse: The converse of implication, which means the right-hand side proposition goes to the left-hand side and vice-versa. It can be written as Q → P.
- Contrapositive: The negation of converse is termed as contrapositive, and it can be represented as ¬ Q → ¬ P.
- Inverse: The negation of implication is called inverse. It can be represented as ¬ P → ¬ Q.

- The most commonly used Rules of Inference are tabulated below

1. Modus Ponens:

- it states that if P and P → Q is true, then we can infer that Q will be true. It can be represented as:

Notation for Modus ponens: $\frac{P \rightarrow Q, \quad P}{\therefore Q}$

Example:

- Statement-1: "If I am sleepy then I go to bed" ==> P → Q
- Statement-2: "I am sleepy" ==> P
- Conclusion: "I go to bed." ==> Q.
- Hence, we can say that, if P → Q is true and P is true then Q will be true.

Proof by Truth table:

P	Q	P → Q	P	Q
1	1	1	1	1
1	0	0	1	No need
0	1	1	0	No need
0	0	1	0	No need

2. Modus Tollens:

- The Modus Tollens rule state that if P → Q is true and ¬ Q is true, then ¬ P will also true. It can be represented as:

Notation for Modus Tollens: $\frac{P \rightarrow Q, \ \sim Q}{\sim P}$

- Statement-1: "If I am sleepy then I go to bed" ==> P → Q
- Statement-2: "I do not go to the bed."==> ~Q
- Statement-3: Which infers that "I am not sleepy" => ~P
- Proof by Truth table:

P	Q	$\sim P$	$\sim Q$	$P \rightarrow Q$
0	**0**	**1**	**1**	**1**
0	**1**	**1**	**0**	**1**
1	**0**	**0**	**1**	**0**
1	**1**	**0**	**0**	**1**

3. Hypothetical Syllogism:

- The Hypothetical Syllogism rule state that if P → R is true whenever P → Q is true, and Q → R is true. It can be represented as the following notation:
- **Example:**
- Statement-1: If you have my home key then you can unlock my home.
  P → Q
  Statement-2: If you can unlock my home then you can take my money. Q → R
  Conclusion: If you have my home key then you can take my money.

P	Q	R	$P \rightarrow Q$	$Q \rightarrow R$	$P \rightarrow R$	
0	0	0	1	1	1	←
0	0	1	1	1	1	←
0	1	0	1	0	1	
0	1	1	1	1	1	←
1	0	0	0	1	1	
1	0	1	0	1	1	
1	1	0	1	0	0	
1	1	1	1	1	1	←

4. Disjunctive Syllogism:

- The Disjunctive syllogism rule state that if P∨Q is true, and ¬P is true, then Q will be true. It can be represented as:

Notation of Disjunctive syllogism: $\frac{P \vee Q,\ \neg P}{Q}$

- Statement-1: Today is Sunday or Monday. ==>P∨Q
- Statement-2: Today is not Sunday. ==> ¬P
- Conclusion: Today is Monday. ==> Q

P	Q	$\neg P$	$P \vee Q$
0	**0**	**1**	**0**
0	**1**	**1**	**1**
1	**0**	**0**	**1**
1	**1**	**0**	**1**

inference rule	tautology	name
p $p \to q$ $\therefore \quad q$	$(p \wedge (p \to q)) \to q$	Modus ponens (mode that affirms)
$\neg q$ $p \to q$ $\therefore \quad \neg p$	$(\neg q \wedge (p \to q)) \to \neg p$	Modus tollens (mode that denies)
$p \to q$ $q \to r$ $\therefore \quad p \to r$	$((p \to q) \wedge (q \to r)) \to (p \to r)$	hypothetical syllogism
$p \vee q$ $\neg p$ $\therefore \quad q$	$((p \vee q) \wedge (\neg p)) \to q$	disjunctive syllogism

5. Addition:

- The Addition rule is one the common inference rule, and it states that If P is true, then P∨Q will be true.

Notation of Addition: $\frac{P}{P \vee Q}$

Example:

- Statement: I have a vanilla ice-cream. ==> P
- Statement-2: I have Chocolate ice-cream.
- Conclusion: I have vanilla or chocolate ice-cream. ==> (P∨Q)

P	Q	P ∨ Q
0	0	0
1	0	1
0	1	1
1	1	1

6. Simplification:

- The simplification rule state that if P∧ Q is true, then Q or P will also be true. It can be represented as:

Notation of Simplification rule: $\frac{P \wedge Q}{Q}$ Or $\frac{P \wedge Q}{P}$

P	Q	$P \wedge Q$
0	0	0
1	0	0
0	1	0
1	1	1

$\therefore \frac{p}{p \vee q}$	$p \rightarrow (p \vee q)$	addition
$\therefore \frac{p \wedge q}{p}$	$(p \wedge q) \rightarrow p$	simplification
$\therefore \frac{\begin{array}{c}p \\ q\end{array}}{p \wedge q}$	$((p) \wedge (q)) \rightarrow (p \wedge q)$	conjunction
$\therefore \frac{\begin{array}{c}p \vee q \\ \neg p \vee r\end{array}}{q \vee r}$	$((p \vee q) \wedge (\neg p \vee r)) \rightarrow (q \vee r)$	resolution

7. Resolution:

- The Resolution rule state that if P∨Q and ¬ P∧R is true, then Q∨R will also be true. **It can be represented as**

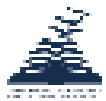
Notation of Resolution $\dfrac{P \vee Q,\ \neg P \wedge R}{Q \vee R}$

P	¬ P	Q	R	$P \vee Q$	¬ P∧R	$Q \vee R$	
0	1	0	0	0	0	0	
0	1	0	1	0	0	1	
0	1	1	0	1	1	1	←
0	1	1	1	1	1	1	←
1	0	0	0	1	0	0	
1	0	0	1	1	0	1	
1	0	1	0	1	0	1	
1	0	1	1	1	0	1	←

Limitations of Propositional logic:

- We cannot represent relations like ALL, some, or none with propositional logic. Example:
  - All the girls are intelligent.
  - Some apples are sweet.
- Propositional logic has limited expressive power.

Propositional logic is a weak language



- Hard to identify “individuals.” E.g., Mary, 3
- Can’t directly talk about properties of individuals or relations between individuals. E.g. “Bill is tall”
- Generalizations, patterns, regularities can’t easily be represented. E.g., all triangles have 3 sides
- First-Order Logic (abbreviated FOL or FOPC) is expressive enough to concisely represent this kind of situation.

  FOL adds relations, variables, and quantifiers, e.g.,

  - *“Every elephant is gray”:* $\forall$ x (elephant(x) $\rightarrow$ gray(x))
  - *“There is a white alligator”:* $\exists$ x (alligator(X) ^ white(X))

- First Order Predicate Logic

First order Predicate Logic

- First-order logic is another way of knowledge representation in AI. It is an extension to propositional logic.
- FOPL is also known as Predicate logic or First-order predicate logic.
- First-order logic is a powerful language that develops information about the objects in a more easy way and can also express the relationship between those objects.

Models for first-order logic

- The models of a logical language are the formal structures that constitute the possible worlds under consideration. Each model links the vocabulary of the logical sentences to elements of the possible world, so that the truth of any sentence can be determined.
- Models have objects in them! The **domain** of a model is DOMAIN the set of objects or **domain elements** it contains. The domain is required to be *nonempty* —every possible world must contain at least one object.
- In the following figure, Richard and John are brothers. Formally speaking, a relation TUPLE is just the set of **tuples** of objects that are related.
- Thus, the brotherhood relation in this model is the set

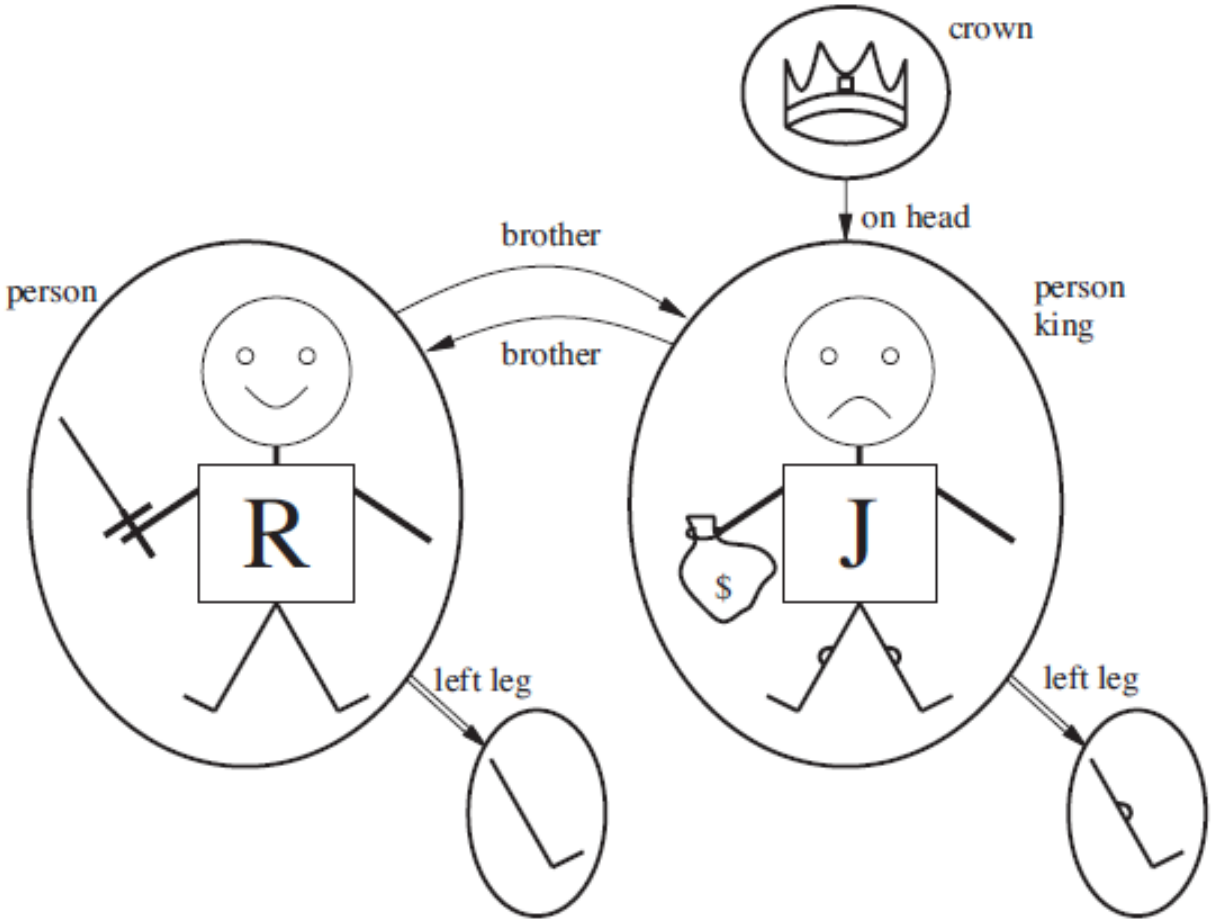
{ Richard the Lionheart, King John, King John, Richard the Lionheart }

- Certain kinds of relationships are best considered as functions, in that a given object must be related to exactly one object in this way. For example, each person has one left leg, so the model has a unary “left leg” function that includes the following mappings:

  Richard the Lionheart → Richard’s left leg

  King John → John’s left leg .

- Strictly speaking, models in first-order logic require **total functions**, that is, there must be a value for every input tuple.



A model containing five objects, two binary relations, three unary relations (indicated by labels on the objects), and one unary function, left-leg.

Symbols and interpretations

FOPL does not only assume that the world contains facts like propositional logic but also assumes the following things in the world like :

- **Objects:** A, B, people, numbers, colors, wars, theories, squares, pits, ……
- **Relations:** It can be unary relation such as: red, round, is adjacent, or n-any relation such as: the sister of, brother of, has color, comes between……
- **Function:** Father of, best friend, third inning of, end of, ...
- As a natural language, first-order logic also has two main parts:
  Syntax
  Semantics

Syntax of First-order Logic: Basic Elements

- **Constant** 1, 2, A, John, Mumbai, cat,....
- **Variables** x, y, z, a, b,....
- **Predicates** Brother, Father, >,....
- **Function** sqrt, LeftLegOf, ...
- **Connectives** ∧, ∨, ¬, ⇒, ⇔
- **Equality** ==
- **Quantifier** ∀, ∃
- We will use various kinds of individual *constants* that denote individuals/objects: **a,b,c, names,…**
- *Individual variables* over objects: ***x ,y ,z , …***
- The *result of applying* a predicate ***P*** to a constant **a** is the proposition ***P*** **(a)**
  Meaning: the object denoted by **a** has the property denoted by **P**.
- if P = “is a prime number”, then P(x) is the propositional form “x is a prime number”.

- In addition to its objects, relations, and functions, each model includes an **interpretation** that specifies exactly which objects, relations and functions are referred to by the constant, predicate, and function symbols
- Example; Richard refers to Richard the Lionheart and John refers to the evil King John.
- Brother refers to the brotherhood relation, that is, the set of tuples of objects.
- OnHead refers to the “on head” relation that holds between the crown and King John; Person, King, and Crown refer to the sets of objects that are persons, kings, and crowns.
- LeftLeg refers to the “left leg” function,

Terms Atomic sentences

- A **term** is a logical expression that refers TERM to an object. Constant symbols are therefore terms, but it is not always convenient to have a distinct symbol to name every object.
- For example, in English we might use the expression “King John’s left leg” rather than giving a name to his leg. This is what function symbols are for: instead of using a constant symbol, we use LeftLeg(John).
- **Atomic sentences** are the most basic sentences of first-order logic.
- These sentences are formed from a predicate symbol followed by a parenthesis with a sequence of terms.
- We can represent atomic sentences as

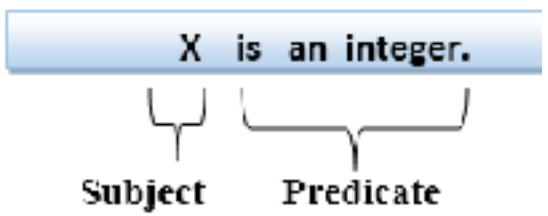
  Predicate (term1, term2, ……, term n).
- Example:

  Ravi and Ajay are brothers: => Brothers(Ravi, Ajay).

  Tom is a cat: => cat (Tom)

Complex Sentences:

- Complex sentences are made by combining atomic sentences using connectives.
- First-order logic statements can be divided into two parts:
  - **Subject**: Subject is the main part of the statement.
  - **Predicate:** A predicate can be defined as a relation, which binds two atoms together in a statement.
- Consider the statement:
- "x is an integer.“
- In the sentence “The dog is sleeping”:
- The phrase “the dog” denotes the subject - which the sentence is about.
- The phrase “is sleeping” denotes the predicate- a property that is true of the subject.
- Predicate logic will follow the same pattern.



Complex Sentences:

- ¬Brother (LeftLeg(Richard), John)
- Brother (Richard , John) ∧ Brother (John,Richard)
- King(Richard) ∨ King(John)
- ¬King(Richard) ⇒ King(John)

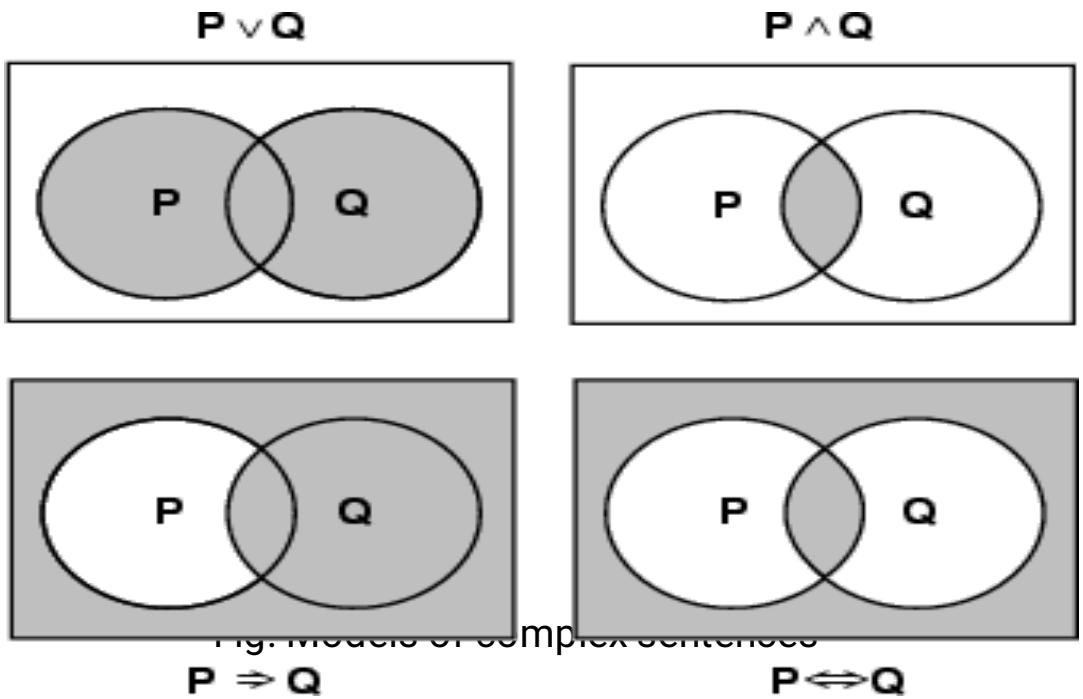


Fig: Models of complex sentences

Examples of Connectives in Use

- "Simon lectures AI and bioinformatics"
  $lectures(simon,ai) \wedge lectures(simon,bioinfo)$
- "If Simon isn't lecturing AI, Bob must be"
  $\neg lectures(simon,ai) \rightarrow lectures(bob,ai)$
- "George and Tony will win or Saddam will lose"
  $(will\_win(george) \wedge will\_win(tony)) \vee \neg will\_win(saddam)$

Example of a Function

- The cost of an omelette at the *red_lion* is $5”
- cost_of is a function
  Input: the name of a meal and the name of a Hotel
  Output: the cost of the meal
  cost_of (omelette, red_lion) = $5
- “Omelettes at the Red Lion cost less than pancakes at House of Pancakes”

less_than (cost_of (omelette,red_lion), cost_of(pancake, pancake_house))

- If we are talking about the output of a function
- We can replace it with the predicate part
- i.e., replace the RHS of equality with the LHS

Quantifiers

- Universal quantifiers are often used with “implies” to form “rules” :

  $(\forall x)$ student(x) $\rightarrow$ smart(x) means “All students are smart”

- Universal quantification is *rarely* used to make blanket statements about every individual in the world:

  $(\forall x)$student(x)$\wedge$smart(x) means “Everyone in the world is a student and is smart”

- **Existential** quantifiers are usually used with “**and**” to specify a list of properties about an individual:

  $(\exists x)$ student(x) $\wedge$ smart(x) means **“There is a student who is smart”**

Quantifiers in First-order logic:

- There are two types of quantifier:

Universal Quantifier(∀)- (for all, everyone, everything)

Specifies that the statement within its range is true for everything or every instance of a particular thing.

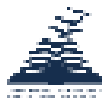
In universal quantifier we use implication **"→".**

Existential quantifier(∃) - (for some, at least one)

Express that the statement within its scope is true for at least one instance of something.

In Existential quantifier we always use **AND or Conjunction symbol (∧)**

Quantifier Scope



- Switching the order of universal quantifiers ***does not*** change the meaning:
  - **(∀x)(∀y)P(x,y) ↔ (∀y)(∀x) P(x,y)**
- Similarly, you can switch the order of existential quantifiers:
  - **(∃x)(∃y)P(x,y) ↔ (∃y)(∃x) P(x,y)**
- Switching the order of universals and existentials ***does*** change meaning:
  - Everyone likes someone: (∀x)(∃y) likes(x,y)
  - Someone is liked by everyone: (∃y)(∀x) likes(x,y)

"∀" is the. FOR ∀LL or*universal* quantifier.
"∃" is the ∃XISTS or*existential* quantifier

For example,

$\forall x\, P(x)$ and $\exists x\, P(x)$ are propositions

- "the parking spaces at University"
  Let P(x) mean "x is full."

∃x P(x) is the proposition saying that

- "Some parking spaces at University are full."
- "There is a parking space at University that is full."
- "At least one parking space at University is full."

- “the parking spaces at University”
  Let P(x) mean “x is occupied”

$\forall$ **x P(x),** is the proposition

- “All parking spaces at University are occupied.”
- “For each parking space at University, that space is full.”

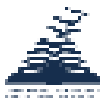
- The universally quantified sentence is equivalent to asserting the following five sentences:
- Richard the Lionheart is a king $\Rightarrow$ Richard the Lionheart is a person.
- King John is a king $\Rightarrow$ King John is a person.
- Richard’s left leg is a king $\Rightarrow$ Richard’s left leg is a person.
- John’s left leg is a king $\Rightarrow$ John’s left leg is a person.
- The crown is a king $\Rightarrow$ the crown is a person

- $\exists$ x Crown(x) $\Rightarrow$ OnHead(x, John) .
- Applying the semantics, the sentence says that at least one of the following assertions is true:
- Richard the Lionheart is a crown $\Rightarrow$ Richard the Lionheart is on John's head;
- King John is a crown $\Rightarrow$ King John is on John's head;
- Richard's left leg is a crown $\Rightarrow$ Richard's left leg is on John's head;

and so on.

- Now an implication is true if both premise and conclusion are true*or if its premise is false* . So if Richard the Lionheart is not a crown, then the first assertion is true and the existential is satisfied.

Combining Quantifiers



$\forall$ x $\exists$ y Loves(x,y)

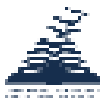
- For everyone (“x”) there is someone (“y”) that they love.

$\exists$y $\forall$ x Loves(x,y)

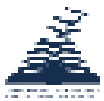
- There is someone (“y”) who is loved by everyone (“X”)

Clearer with parentheses: $\exists$ y ($\forall$ x Loves(x,y))

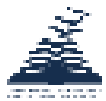
	whether it is a negation, conjunction, disjunction, implication, universal formula, existential formula	Scope of The Quantizer	Free Variables	Whether it is a sentence
1. $\exists x(A(x, y) \wedge B(x))$				
2. $\exists x(\exists y(A(x, y) \rightarrow B(x))$				
3. $\neg\exists x(\exists y(A(x, y))) \rightarrow B(x)$				
4. $\forall x(\neg\exists y(A(x, y)))$				
5. $\exists x(A(x, y)) \wedge B(x)$				
6. $\exists x(A(x, x)) \wedge \exists y(B(y))$				



	whether it is a negation, a conjunction, a disjunction, an implication, a universal formula, or an existential formula	Scope of The Quantizer	Free Variables	Whether it is a sentence
1. ∃x(A(x, y) ∧ B(x))	Existential	∃x: (A(x, y) ∧ B(x))	y	No
2. ∃x(∃y(A(x, y) → B(x))	Existential	∃x: (∃y(A(x, y) → B(x)) ∃y: A(x,y)	None	Yes
3. ¬∃x(∃y(A(x, y))) → B(x)	Implication	∃x: (∃y(A(x, y)) ∃y: A(x, y)	x in B(x)	No
4. ∀x(¬∃y(A(x, y)))	Universal	∀x : (¬∃y(A(x, y))) ∃y: A(x, y)	None	Yes
5. ∃x(A(x, y)) ∧ B(x)	Conjunction	∃x: A(x, y))	y, x in B(x)	No
6. ∃x(A(x, x)) ∧ ∃y(B(y))	Conjunction	∃x: (A(x, x)) ∃y: (B(y))	None	Yes



Translating English to FOL



Every gardener likes the sun.

$\forall x$ gardener(x) $\rightarrow$ likes(x,Sun)

You can fool some of the people all of the time.

$\exists x \forall t$ person(x) $\wedge$time(t) $\rightarrow$ can-fool(x,t)

You can fool all of the people some of the time.

$\forall x \exists t$ (person(x) $\rightarrow$ time(t) $\wedge$can-fool(x,t))

$\forall x$ (person(x) $\rightarrow \exists t$ (time(t) $\wedge$can-fool(x,t)))

All purple mushrooms are poisonous.

$\forall x$ (mushroom(x) $\wedge$ purple(x)) $\rightarrow$ poisonous(x)

No purple mushroom is poisonous.

$\neg \exists x$ purple(x) $\wedge$ mushroom(x) $\wedge$ poisonous(x)

$\forall x$ (mushroom(x) $\wedge$ purple(x)) $\rightarrow \neg$poisonous(x)

There are exactly two purple mushrooms.

$\exists x \exists y$ mushroom(x) $\wedge$ purple(x) $\wedge$ mushroom(y) $\wedge$ purple(y) ^ $\neg$(x=y) $\wedge \forall z$ (mushroom(z) $\wedge$ purple(z)) $\rightarrow$ ((x=z) $\vee$ (y=z))

Clinton is not tall.

$\neg$tall(Clinton)

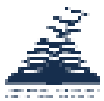
X is above Y iff X is on directly on top of Y or there is a pile of one or more other objects directly on top of one another starting with X and ending with Y.

$\forall x \forall y$ above(x,y) $\leftrightarrow$ (on(x,y) $\vee \exists z$ (on(x,z) $\wedge$ above(z,y)))

Translate the following sentences into FOL

1. Everything is bitter or sweet
2. Either everything is bitter or everything is sweet
3. There is somebody who is loved by everyone
4. Nobody is loved by no one
5. If someone is noisy, everybody is annoyed
6. Frogs are green.
7. Frogs are not green.
8. No frog is green.
9. Some frogs are not green.
10. A mechanic likes Bob.
11. A mechanic likes herself.
12. Every mechanic likes Bob.
13. Some mechanic likes every nurse.
14. There is a mechanic who is liked by every nurse.

1. $\forall x(B(x) \vee S(x))$
2. $\forall x(B(x) \vee \forall x(S(x))$
3. $\exists x(\forall y(L(y,x)))$
4. $\neg\exists x(\neg\exists y(L(y,x)))$
5. $\exists x(N(x) \rightarrow \forall y(A(y)))$
6. $\forall x(Fx \rightarrow Gx)$
7. $\forall x(Fx \rightarrow \neg Gx) = \neg\exists x(Fx \wedge Gx)$
8. $\neg\exists x(Fx \wedge Gx) = \forall x(Fx \rightarrow Gx)$
9. $\exists x(Fx \wedge \neg Gx)$
10. $\exists X(Mx \wedge L(x,b))$
11. $\exists x(Mx \wedge L(x,x))$
12. $\forall x(Mx \rightarrow L(x,b))$
13. $\exists x(Mx \wedge \forall y(Ny \rightarrow L(x,y)))$
14. $\exists x(Mx \wedge \forall y(Ny \rightarrow L(y,x)))$



Nested quantifiers

- "Brothers are siblings" can be written as

  $\forall x \forall y \text{ Brother}(x, y) \Rightarrow \text{Sibling}(x, y)$.

- "siblinghood is a symmetric relationship", we can write

  $\forall x, y \text{ Sibling}(x, y) \Leftrightarrow \text{Sibling}(y, x)$.

- "Everybody loves somebody" means that for every person, there is someone that person loves:

  $\forall x \exists y \text{ Loves}(x, y)$.

- On the other hand, to say "There is someone who is loved by everyone," we write

  $\exists y \forall x \text{ Loves}(x, y)$.

[Note:-The order of quantification is very important. It becomes clearer if we insert parentheses.

$\forall x (\exists y \text{ Loves}(x, y))$ says that everyone has a particular property, namely, the property that they love someone. On the other hand, $\exists y (\forall x \text{ Loves}(x, y))$ says that someone in the world has a particular property, namely the property of being loved by everybody]

Connections between ∀ and ∃

- The two quantifiers are actually intimately connected with each other, through negation.
- Asserting that everyone dislikes parsnips is the same as asserting there does not exist someone who likes them, and vice versa:

  ∀ x ¬Likes(x, Parsnips) is equivalent to ¬∃ x Likes(x, Parsnips) .
- “Everyone likes ice cream” means that there is no one who does not like ice cream:

  ∀ x Likes(x, IceCream) is equivalent to ¬∃ x ¬Likes(x, IceCream) .
- Because ∀ is really a conjunction over the universe of objects and ∃ is a disjunction, they obey De Morgan’s rules. The De Morgan rules for quantified and unquantified sentences are as follows:
- ∀ x ¬P ≡ ¬∃x P ¬(P ∨ Q) ≡ ¬P ∧¬Q
- ¬∀x P ≡ ∃x ¬P ¬(P ∧ Q) ≡ ¬P ∨¬Q
- ∀x P ≡ ¬∃x ¬P P∧ Q ≡ ¬(¬P ∨¬Q)
- ∃x P ≡ ¬∀x ¬P P∨ Q ≡ ¬(¬P ∧¬Q) .

Equality

- We can use the **equality symbol** to signify that two terms refer to the same object. For example,

  Father (John)=Henry

  says that the object referred to by Father (John) and the object referred to by Henry are the same. Because an interpretation fixes the referent of any term, determining the truth of an equality sentence is simply a matter of seeing that the referents of the two terms are the same object.

- It can also be used with negation to insist that two terms are not the same object. To say that Richard has at least two brothers, we would write

  $\exists$ x, y Brother (x,Richard) $\wedge$ Brother (y,Richard) $\wedge \neg$(x=y) .

- The sentence

  $\exists$ x, y Brother (x,Richard) $\wedge$ Brother (y,Richard)

  does not have the intended meaning

Variables

- "There's a meal at the Red Lion which costs $3" (more expressive now)
- *cost_of(meal,red_lion) = 3*
- We're really talking about some general meal
  - Not a meal in particular
  - Call this meal X (a variable)
- *cost_of (X, red_lion) = 3*

need to express "There is such a meal X". This is represented as: ∃

The exists sign is known as a existential quantifier

- *∃ X (cost_of (X, red_lion) = 3)*

Examples

- "There is a meal at the Red Lion which costs three pounds"

$$\exists X\ (meal(X) \wedge cost\_of\ (red\_lion, X) = 3)$$

- "All the meals at the Red Lion cost three pounds"

Replace exists by forall:

$$\forall X\ (meal(X) \wedge cost\_of\ (red\_lion, X) = 3)$$

$$\forall X\ (meal(X) \rightarrow cost\_of(red\_lion, X) = 3)$$

- "Every Monday and Wednesday I go to John's house for dinner"

$$\forall X\ (day(X,mon) \vee day(X,weds)$$

$$\rightarrow go(me, house(john)) \wedge eat(me,dinner))$$

“All things in the bag are red”

Which of these translations is correct?

1. $\exists X\ (in\_bag(X) \rightarrow red(X))$
2. $\forall X\ (red(X) \rightarrow in\_bag(X))$
3. $\forall X\ (\forall Y\ (bag(X) \wedge in\_bag(Y,X) \rightarrow red(Y)))$

USING FIRST-ORDER LOGIC

Assertions and queries in first-order logic

- Sentences are added to a knowledge base using TELL, exactly as in propositional logic. Such sentences are called **assertions**. For example, we can assert that John is a king, Richard is a person, and all kings are persons:
    - TELL(KB, King(John)) .
    - TELL(KB, Person(Richard)) .
    - TELL(KB, $\forall$ x King(x) $\Rightarrow$ Person(x)) .

- We can ask questions of the knowledge base using ASK. For example

  ASK(KB, King(John))

returns true. Questions asked with ASK are called **queries** or **goals**. Generally speaking, any query that is logically entailed by the knowledge base should be answered affirmatively.

USING FIRST-ORDER LOGIC

For example, given the two preceding assertions, the query ASK(KB, Person(John))

- should also return true. We can ask quantified queries, such as

  ASK(KB, ∃ x Person(x))

- The answer is true, but this is perhaps not as helpful as we would like. It is rather like answering “Can you tell me the time?” with “Yes.” If we want to know what value of x makes the sentence true, we will need a different function, ASKVARS, which we call with

  ASKVARS(KB, Person(x))

- In this case there will be two answers: {x/John} and {x/Richard}. Such an answer is called a **substitution** or **binding list**.

Practical Applications of Predicate Logic

- Basis for proving correctness of computer programs
- Basis for many Artificial Intelligence systems.
- Predicate-logic like statements are supported by some sophisticated database query engines