UNIT 3

UNIT 3

WHAT IS KNOWLEDGE REPRESENTATION?

Humans are best at understanding, reasoning, and interpreting knowledge. Human knows things, which is knowledge and as per their knowledge they perform various actions in the real world. But how machines do all these things comes under knowledge representation and reasoning. Hence we can describe Knowledge representation as following:

- Knowledge representation and reasoning (KR, KRR) is the part of Artificial intelligence which concerned with AI agents thinking and how thinking contributes to intelligent behavior of agents.
- It is responsible for representing information about the real world so that a computer can understand and can utilize this knowledge to solve the complex real world problems such as diagnosis a medical condition or communicating with humans in natural language.
- It is also a way which describes how we can represent knowledge in artificial intelligence. Knowledge representation is not just storing data into some database, but it also enables an intelligent machine to learn from that knowledge and experiences so that it can behave intelligently like a human.

What to Represent:

- **Object:** All the facts about objects in our world domain. E.g., Guitars contains strings, trumpets are brass instruments.
- **Events:** Events are the actions which occur in our world.
- **Performance:** It describe behaviour which involves knowledge about how to do things.
- **Meta-knowledge:** It is knowledge about what we know.
- **Facts:** Facts are the truths about the real world and what we represent.
- **Knowledge-Base:** The central component of the knowledge-based agents is the knowledge base. It is represented as KB. The Knowledgebase is a group of the Sentences.

Knowledge: *Knowledge is awareness or familiarity gained by experiences of facts, data, and situations.*

TYPES OF KNOWLEDGE

1. Declarative Knowledge:

- Declarative knowledge is to know about something.
- It includes concepts, facts, and objects.
- It is also called descriptive knowledge and expressed in declarativesentences.
- It is simpler than procedural language.

2. Procedural Knowledge

- It is also known as imperative knowledge.
- Procedural knowledge is a type of knowledge which is responsible for knowing how to do something.
- It can be directly applied to any task.
- It includes rules, strategies, procedures, agendas, etc.
- Procedural knowledge depends on the task on which it can be applied.

3. Meta-knowledge:

- Knowledge about the other types of knowledge is called Meta-knowledge.

4. Heuristic knowledge:

- Heuristic knowledge is representing knowledge of some experts in a filed or subject.
- Heuristic knowledge is rules of thumb based on previous experiences, awareness of approaches, and which are good to work but not guaranteed.

5. Structural knowledge:

- Structural knowledge is basic knowledge to problem-solving.
- It describes relationships between various concepts such as kind of, part of, and grouping of something.
- It describes the relationship that exists between concepts or objects.

Properties a Good Knowledge Representation System Should Have

1. **Representational adequacy** : It should be able to represent the different kinds of knowledge required. A major property of a knowledge representation system is that it is adequate and can make an AI system understand, i.e., represent all the knowledge required by it to deal with a particular field or domain.

2. **Inferential adequacy** : The KR system should be able to come up with new structures or knowledge that it can infer from the original or existing structures. The knowledge representation system is flexible enough to deal with the present knowledge to make way for newly possessed knowledge.

3. **Inferential efficiency** : It should be able to integrate additional mechanisms to existing knowledge structures to direct them toward promising directions. The representation system cannot accommodate new knowledge in the presence of the old knowledge, but it can add this knowledge efficiently and in a seamless manner.

4. **Acquisitional efficiency** : The ability to acquire the new knowledge easily using automatic methods. The final property of the knowledge representation system will be its ability to gain new knowledge automatically, helping the AI to add to its current knowledge and consequently become increasingly smarter and productive.

To date, no single KR system has all of these properties.

AI KNOWLEDGE CYCLE:

An Artificial intelligence system has the following components for displaying intelligent behaviour:

- Perception
- Learning
- Knowledge Representation and Reasoning
- Planning
- Execution

APPROACHES TO KNOWLEDGE REPRESENTATION:

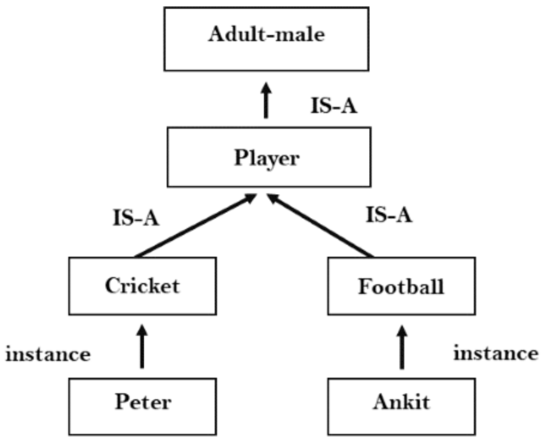
1. Simple relational knowledge:

- It is the simplest way of storing facts which uses the relational method, and each fact about a set of the object is set out systematically in columns.
- This approach of knowledge representation is famous in database systems where the relationship between different entities is represented.

- ❖ This approach has little opportunity for inference.

2. Inheritable knowledge:

- ❖ In the inheritable knowledge approach, all data must be stored into a hierarchy of classes.
- ❖ All classes should be arranged in a generalized form or a hierarchal manner.
- ❖ In this approach, we apply inheritance property.
- ❖ Elements inherit values from other members of a class.
- ❖ This approach contains inheritable knowledge which shows a relation between instance and class, and it is called instance relation.
- ❖ Every individual frame can represent the collection of attributes and its value.
- ❖ In this approach, objects and values are represented in Boxed nodes.
- ❖ We use Arrows which point from objects to their values.
- ❖ **Example:**



3. Inferential knowledge:

- ❖ Inferential knowledge approach represents knowledge in the form of formal logics.
- ❖ This approach can be used to derive more facts.
- ❖ It guaranteed correctness.
- ❖ **Example:** Let's suppose there are two statements:
  - o Marcus is a man
  - o All men are mortal

    Then it can represent as;

    man(Marcus)

    ∀x = man (x) ----------> mortal (x)s

4. Procedural knowledge:

- ❖ Procedural knowledge approach uses small programs and codes which describes how to do specific things, and how to proceed.
- ❖ In this approach, one important rule is used which is **If-Then rule**.
- ❖ In this knowledge, we can use various coding languages such as **LISP language** and **Prolog language**.
- ❖ We can easily represent heuristic or domain-specific knowledge using this approach.
- ❖ But it is not necessary that we can represent all cases in this approach.

PREDICATE LOGIC

https://www.javatpoint.com/predicate-logic

- A predicate is **an expression of one or more variables determined on some specific domain**.
- A predicate with variables can be made a proposition by either authorizing a value to the variable or by quantifying the variable.
- Predicate logic is **a mathematical model that is used for reasoning with predicates**. Predicates are functions that map variables to truth values.
- They are essentially boolean functions whose value could be true or false, depending on the arguments to the predicate.
- They are generalizations of propositional variables.

The following are some examples of predicates.

- Consider E(x, y) denote "x = y"
- Consider X(a, b, c) denote "a + b + c = 0"
- Consider M(x, y) denote "x is married to y."

Quantifier:

The variable of predicates is quantified by quantifiers. There are two types of quantifier in predicate logic - **Existential Quantifier** and **Universal Quantifier.**

(a) Existential Quantifier:

- ✓ If p(x) is a proposition over the universe U. Then it is denoted as ∃x p(x) and read as "There exists at least one value in the universe of variable x such that p(x) is true. The quantifier ∃ is called the existential quantifier.
- ✓ There are several ways to write a proposition, with an existential quantifier, i.e., (∃x∈A)p(x) or ∃x∈A such that p (x) or (∃x)p(x) or p(x) is true for some x ∈A.

(b) Universal Quantifier:

- ✓ If p(x) is a proposition over the universe U. Then it is denoted as ∀x,p(x) and read as "For every x∈U,p(x) is true." The quantifier ∀ is called the Universal Quantifier.
- ✓ There are several ways to write a proposition, with a universal quantifier. ∀x∈A,p(x) or p(x), ∀x ∈A Or ∀x,p(x) or p(x) is true for all x ∈A.

Negation of Quantified Propositions:

- ✓ When we negate a quantified proposition, i.e., when a universally quantified proposition is negated, we obtain an existentially quantified proposition,and when an existentially quantified proposition is negated, we obtain a universally quantified proposition.
- ✓ The two rules for negation of quantified proposition are as follows. These are also called DeMorgan's Law.

Example: Negate each of the following propositions:

1.$\forall x\ p(x) \wedge \exists\ y\ q(y)$

Sol: $\sim.\forall x\ p(x) \wedge \exists\ y\ q(y))$

$\cong \sim\forall\ x\ p(x) \vee \sim\exists y q\ (y)$ $\quad (\therefore \sim(p\wedge q) = \sim p \vee \sim q)$

$\cong \exists\ x \sim p(x) \vee \forall y \sim q(y)$

2. $(\exists x \in U)\ (x+6=25)$

Sol: $\sim(\exists\ x \in U)\ (x+6=25)$

$\cong \forall\ x \in U \sim\ (x+6)=25$

$\cong (\forall\ x \in U)\ (x+6) \neq 25$

3. $\sim(\exists\ x\ p(x) \vee \forall\ y\ q(y)$

Sol: $\sim(\exists\ x\ p(x) \vee \forall\ y\ q(y))$

$\cong \sim\exists\ x\ p(x) \wedge \sim\forall\ y\ q(y)$ $\quad (\therefore \sim(p \vee q) = \sim p \wedge \sim q)$

$\cong \forall\ x \sim\ p(x) \wedge \exists y \sim q(y))$

Propositions with Multiple Quantifiers:

- ✓ The proposition having more than one variable can be quantified with multiple quantifiers.
- ✓ The multiple universal quantifiers can be arranged in any order without altering the meaning of the resulting proposition.
- ✓ Also, the multiple existential quantifiers can be arranged in any order without altering the meaning of the proposition.
- ✓ The proposition which contains both universal and existential quantifiers, the order of those quantifiers can't be exchanged without altering the meaning of the proposition, e.g., the proposition $\exists x\ \forall\ y\ p(x,y)$ means "There exists some x such that p (x, y) is true for every y."

Example: Write the negation for each of the following. Determine whether the resulting statement is true or false. Assume U = R.

1. $\forall\ x\ \exists\ m(x^2<m)$

Sol: Negation of $\forall\ x\ \exists\ m(x^2<m)$ is $\exists\ x\ \forall\ m\ (x^2 \geq m)$. The meaning of $\exists\ x\ \forall\ m\ (x^2 \geq m)$ is that there exists for some x such that $x^2 \geq m$, for every m. The statement is true as there is some greater x such that $x^2 \geq m$, for every m.

2. $\exists\ m \forall\ x(x^2<m)$

Sol: Negation of $\exists\ m\ \forall\ x\ (x^2<m)$ is $\forall\ m \exists x\ (x^2 \geq m)$. The meaning of $\forall\ m \exists x\ (x^2 \geq m)$ is that for every m, there exists for some x such that $x^2 \geq m$. The statement is true as for every m, there exists for some greater x such that $x^2 \geq m$.

FIRST-ORDER LOGIC:

https://www.javatpoint.com/first-order-logic-in-artificial-intelligence

- ✓ First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic.
- ✓ FOL is sufficiently expressive to represent the natural language statements in a concise way.

- ✓ First-order logic is also known as **Predicate logic or First-order predicate logic (FOPL)**.
- ✓ 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.
- ✓ First-order logic (like natural language) does not only assume that the world contains facts like propositional logic but also assumes the following things in the world:
    - **Objects:** A, B, people, numbers, colors, wars, theories, squares, pits, wumpus
    - **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**

Points to remember:

- ✓ The main connective for universal quantifier ∀ is implication →.
- ✓ The main connective for existential quantifier ∃ is and ∧.

Properties of Quantifiers:

- ✓ In universal quantifier, ∀x∀y is similar to ∀y∀x.
- ✓ In Existential quantifier, ∃x∃y is similar to ∃y∃x.
- ✓ ∃x∀y is not similar to ∀y∃x.

Some Examples of FOL using quantifier:

1. All birds fly.
In this question the predicate is "**fly(bird)**."
And since there are all birds who fly so it will be represented as follows.

∀x bird(x) →fly(x).

2. Every man respects his parent.
In this question, the predicate is "**respect(x, y)," where x=man, and y= parent**.
Since there is every man so will use ∀, and it will be represented as follows:

∀x man(x) → respects (x, parent).

3. Some boys play cricket.
In this question, the predicate is "**play(x, y)**," where x= boys, and y= game. Since there are some boys so we will use ∃**, and it will be represented as**:

∃x boys(x) → play(x, cricket).

4. Not all students like both Mathematics and Science.
In this question, the predicate is "**like(x, y)," where x= student, and y= subject**.
Since there are not all students, so we will use ∀ **with negation, so** following representation for this:

¬∀ (x) [student(x) → like(x, Mathematics) ∧ like(x, Science)].

FREE AND BOUND VARIABLES:

The quantifiers interact with variables which appear in a suitable way. There are two types of variables in First-order logic which are given below:

(1) Free Variable: A variable is said to be a free variable in a formula if it occurs outside the scope of the quantifier.

Example: ∀x ∃(y)[P (x, y, z)], where z is a free variable.

(2) Bound Variable: A variable is said to be a bound variable in a formula if it occurs within the scope of the quantifier.

Example: ∀x [A (x) B(y)], here x and y are the bound variables.