Knowledge Representation

Knowledge Representation:

- Knowledge representation is a subject in cognitive science as well as in artificial intelligence and knowledge modeling.
- In cognitive science it is concerned with how people store and process information.
- In artificial intelligence (AI) and knowledge modeling (KM) it is a way to store knowledge so that programs can process it and use it, for example to support computer-aided design or to emulate human intelligence.
- AI researchers have borrowed representation theories from cognitive science.

Knowledge Representation Issues

- The aim is to show how logic can be used to form representations of the world and how a process of inference can be used to derive new representations about the world and how these can be used by an intelligent agent to deduce what to do.

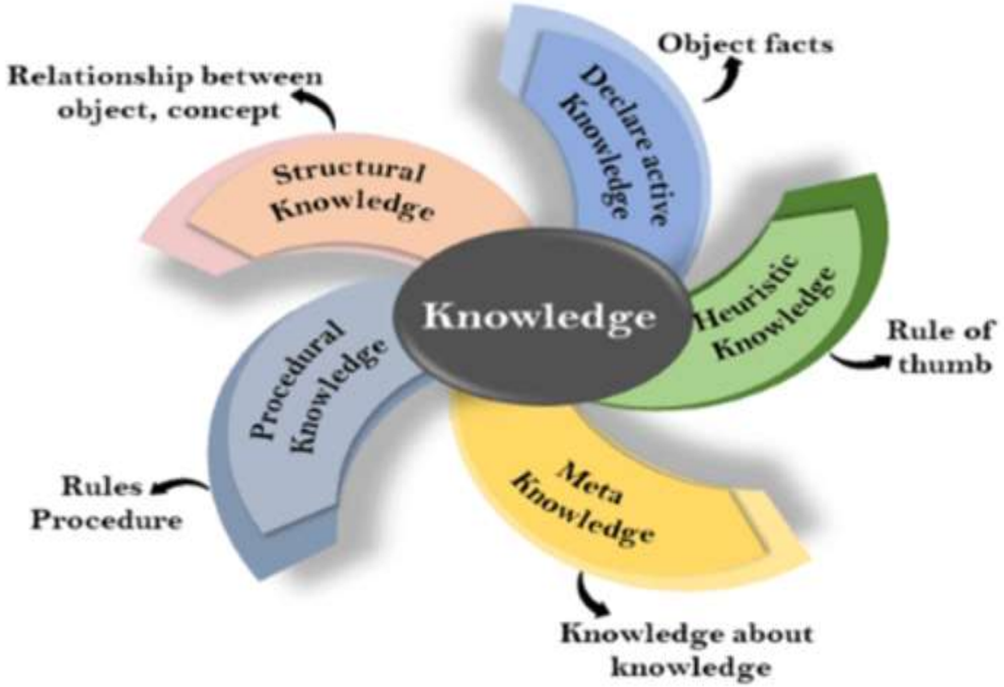
We require:

- A *formal language* to represent knowledge in a computer tractable form.
- *Reasoning* - Processes to manipulate this knowledge to deduce non-obvious facts.

- **Knowledge:** Knowledge is awareness or familiarity gained by experiences of facts, data, and situations. Following are the types of knowledge in artificial intelligence:

Types of knowledge

- Following are the various types of knowledge:



1. Declarative Knowledge:

- Declarative knowledge is to know about something.
- It includes concepts, facts, and objects.

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 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:

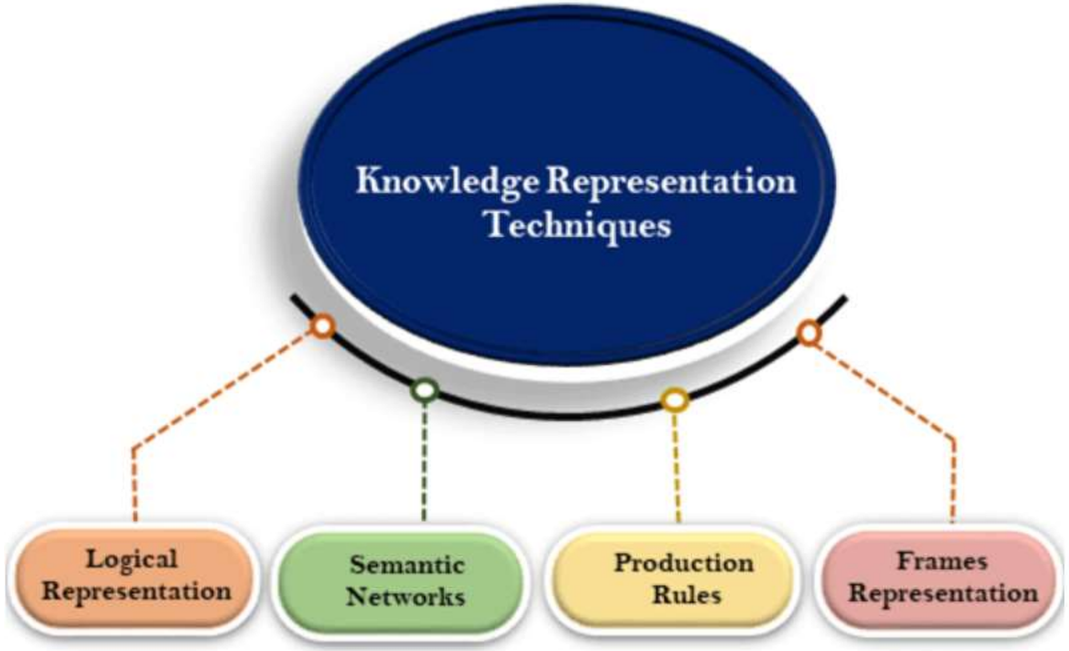
- Heuristic knowledge is representing knowledge of some experts in a filed or subject.

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.

- There are mainly four ways of knowledge representation which are given as follows

1. Logical Representation
2. Semantic Network Representation
3. Frame Representation
4. Production Rules



Why logic?

- The challenge is to design a language which allows one to represent all the necessary knowledge.
- **Logic** makes statements about the world which are true (or false) if the state of affairs it represents is the case (or not the case).
- Compared to natural languages (expressive but context sensitive) and programming languages (good for concrete data structures but not expressive) logic combines the advantages of natural languages and formal languages.

Logical Representation

- Logical representation is a language with some concrete rules which deals with propositions and has no ambiguity in representation.
- Logical representation means drawing a conclusion based on various conditions.
- This representation lays down some important communication rules.
- It consists of precisely defined syntax and semantics which supports the sound inference.
- Each sentence can be translated into logics using syntax and semantics.

Syntax:

- Syntaxes are the rules which decide how we can construct legal sentences in the logic.
- It determines which symbol we can use in knowledge representation.
- How to write those symbols.

Semantics:

- Semantics are the rules by which we can interpret the sentence in the logic.
- Semantic also involves assigning a meaning to each sentence.

Logical representation can be categorized into mainly two logics:

1. Propositional Logics
2. Predicate logics

Propositional logic (PL)

- 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.

Example:

a) It is Sunday.

b) The Sun rises from West (False proposition)

c) 3+3= 7(False proposition)

d) 5 is a prime number.

Following are some basic facts about propositional logic:

- Propositional logic is also called Boolean logic as it works on 0 and 1.
- In propositional logic, we use symbolic variables to represent the logic, such A, B, C, P, Q, R, etc.
- Propositions can be either true or false, but it cannot be both.
- Propositional logic consists of an object, relations or function, and **logical connectives**.
- Connectives can be said as a logical operator which connects two sentences.
- A proposition formula which is always true is called **tautology**, and it is also called a valid sentence.
- A proposition formula which is always false is called **Contradiction**.
- 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.

Syntax of propositional logic:

- The syntax of propositional logic defines the allowable sentences for the knowledge representation. There are two types of Propositions:
- **Atomic Propositions**
- **Compound propositions**

- **Atomic Proposition:** Atomic propositions are the simple propositions. It consists of a single proposition symbol. These are the sentences which must be either true or false.

Example:

a) 2+2 is 4, it is an atomic proposition as it is a **true** fact.

b) "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:

a) "It is raining today, and street is wet."

b) "Ankit is a doctor, and his clinic is in Mumbai."

- **Logical Connectives:**
- Logical connectives are used to connect two simpler propositions or representing a sentence logically. We can create compound propositions with the help of logical connectives.
- There are mainly five connectives, which are given as follows:
- **Negation:** A sentence such as ¬ P is called negation of P. A literal can be either Positive literal or negative literal.

- **Conjunction:** A sentence which has ∧ connective such as, **P ∧ Q** is called a conjunction.
  Example: Rohan is intelligent and hardworking. It can be written as,
  P= Rohan is intelligent,
  Q= Rohan is hardworking. → P∧ Q.

- **Disjunction:** A sentence which has ∨ connective, such as **P ∨ Q**. is called disjunction, where P and Q are the propositions.
  Example: "Ritika is a doctor or Engineer",
  Here P= Ritika is Doctor. Q= Ritika is Doctor, so we can write it as **P ∨ Q**.

- **Implication:** A sentence such as P → Q, is called an implication. Implications are also known as if-then rules. It can be represented as
  If it is raining, then the street is wet.
  Let P= It is raining, and Q= Street is wet, so it is represented as P → Q

- **Biconditional:** A sentence such as **$P \Leftrightarrow Q$ is a Biconditional sentence, example If I am breathing, then I am alive**

  P= I am breathing, Q= I am alive, it can be represented as $P \Leftrightarrow Q$.

Precedence of connectives:

- Parenthesis, Negation, Conjunction(AND), Disjunction(OR), Implication, Biconditional

Rules of Inference

Inference:

- 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.**

Inference rules:

- 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.
- In inference rules, the implication among all the connectives plays an important role. Following are some terminologies related to inference rules:

- **Implication:** It is one of the logical connectives which can be represented as $P \rightarrow Q$. 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 $P \rightarrow Q$ & $Q \rightarrow P$.
- **Contrapositive:** The negation of converse is termed as contrapositive, and it can be represented as $\neg Q \rightarrow \neg P$.
- **Inverse:** The negation of implication is called inverse. It can be represented as $\neg P \rightarrow \neg Q$.

Types of Inference rules:

1. Modus Ponens:

- The Modus Ponens rule is one of the most important rules of inference, and 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: $\dfrac{P \rightarrow Q, \quad P}{\therefore Q}$

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.

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: $\dfrac{P \to Q,\ \sim Q}{\sim P}$

Example

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

3. Hypothetical Syllogism:

- The Hypothetical Syllogism rule state that if $P \rightarrow R$ is true whenever $P \rightarrow Q$ is true, and $Q \rightarrow R$ 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. $\mathbf{P \rightarrow Q}$
Statement-2: If you can unlock my home then you can take my money. $\mathbf{Q \rightarrow R}$
Conclusion: If you have my home key then you can take my money. $\mathbf{P \rightarrow R}$

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: $\frac{P \lor Q, \quad \neg P}{Q}$

- **Example:**

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

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: $\dfrac{P}{P \lor Q}$

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:a

Notation of Simplification rule: $\dfrac{P \land Q}{Q}$ Or $\dfrac{P \land Q}{P}$

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**

$$\text{Notation of Resolution} \frac{P \lor Q, \quad \neg P \land R}{Q \lor R}$$

First-Order logic:

- First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic.
- First-order logic 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.

- 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**

Syntax of First-Order logic:

- The syntax of FOL determines which collection of symbols is a logical expression in first-order logic. The basic syntactic elements of first-order logic are symbols. We write statements in short-hand notation in FOL.

Basic Elements of First-order logic:

- Following are the basic elements of FOL syntax:

Constant	1, 2, A, John, Mumbai, cat,....
Variables	x, y, z, a, b,....
Predicates	Brother, Father, >,....
Function	sqrt, LeftLegOf, ...
Connectives	∧, ∨, ¬, ⇒, ⇔
Equality	==
Quantifier	∀, ∃

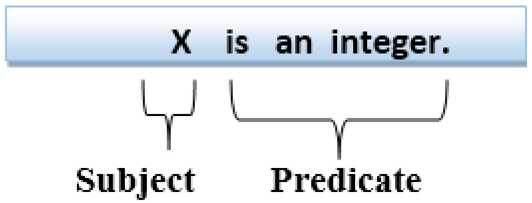
Atomic sentences:

- 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). Chinky is a cat: => cat (Chinky)**.

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.**", it consists of two parts, the first part **x** is the subject of the statement and second part "**is an integer**," is known as a predicate.



- All Boys like cricket

predicate ← like(boys,cricket) → parameters

- Some boys like cricket

like(boys,cricket)

Quantifiers in First-order logic:

- A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimen in the universe of discourse.
- These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression. There are two types of quantifier:

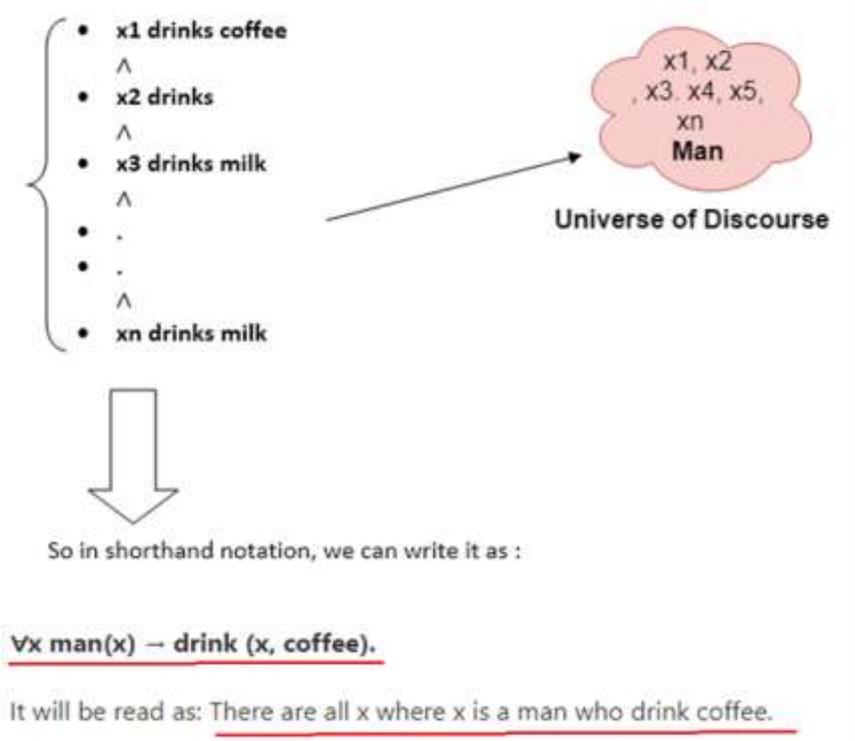
  - **Universal Quantifier, (for all, everyone, everything)**
  - **Existential quantifier, (for some, at least one).**

Universal Quantifier:

- Universal quantifier is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing.
- The Universal quantifier is represented by a symbol ∀, which resembles an inverted A.
- If x is a variable, then ∀x is read as:
- **For all x**
- **For each x**
- **For every x**

Example:

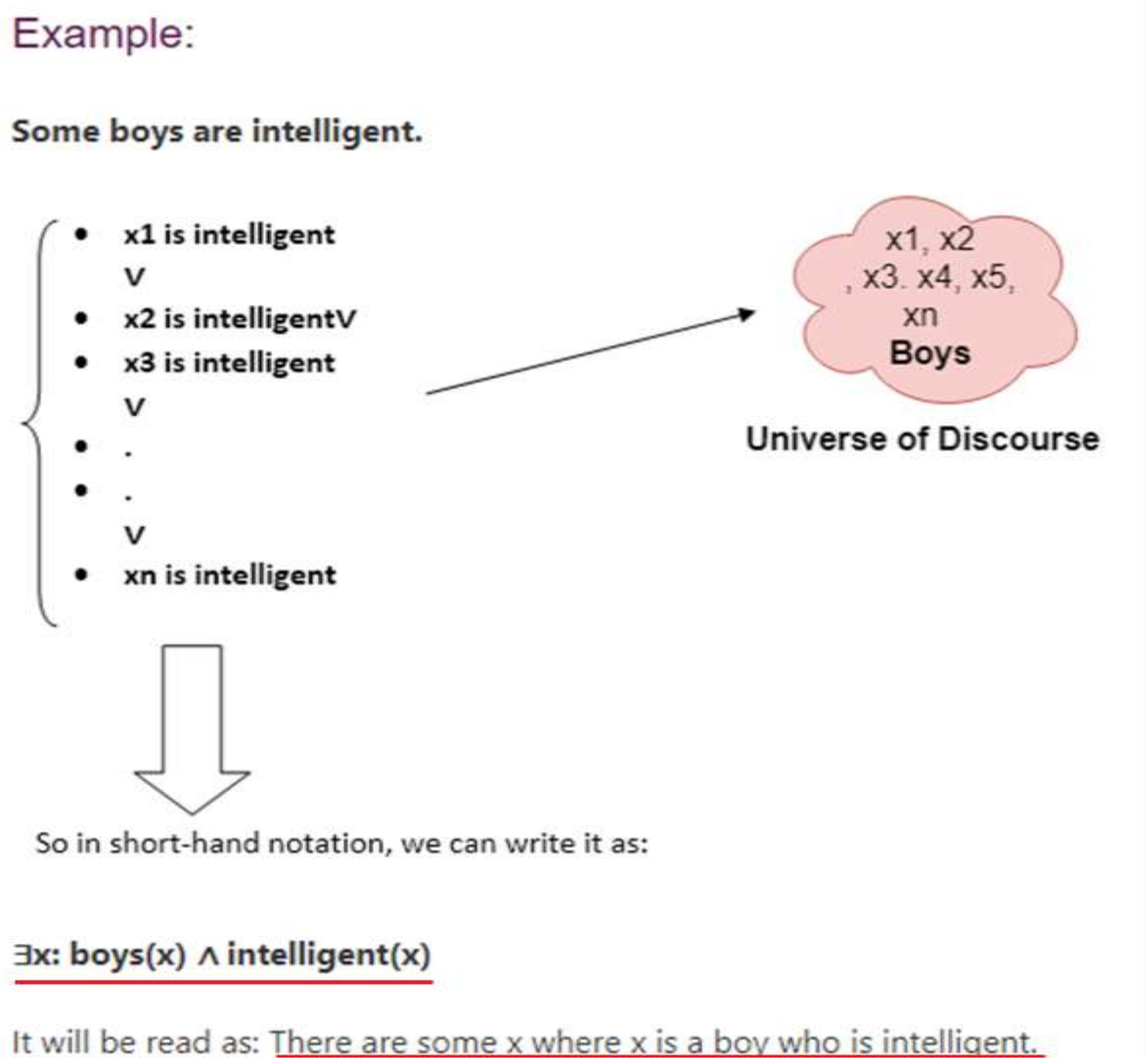
- **All man drink coffee.**



Existential Quantifier:

- Existential quantifiers are the type of quantifiers, which express that the statement within its scope is true for at least one instance of something.
- It is denoted by the logical operator ∃, which resembles as inverted E. When it is used with a predicate variable then it is called as an existential quantifier.

- If x is a variable, then existential quantifier will be ∃x or ∃(x). And it will be read as:
- **There exists a 'x.'**
- **For some 'x.'**
- **For at least one 'x.'**



Properties of Quantifiers:

- In universal quantifier, $\forall x \forall y$ is similar to $\forall y \forall x$.
- In Existential quantifier, $\exists x \exists y$ is similar to $\exists y \exists x$.
- $\exists x \forall y$ is not similar to $\forall y \exists x$.

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.

$$\forall x \; bird(x) \rightarrow fly(x).$$

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:

$$\forall x \; man(x) \rightarrow respects\,(x, parent).$$

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**:

$$\exists x \; boys(x) \rightarrow play(x, cricket).$$

- **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:
  $$\neg\forall (x)\ [\ student(x) \rightarrow like(x, Mathematics) \land like(x, Science)].$$
- **5. Only one student failed in Mathematics.**
  In this question, the predicate is "**failed(x, y)," where x= student, and y= subject**.
  Since there is only one student who failed in Mathematics, so we will use following representation for this:
  $$\exists(x)\ [\ student(x) \rightarrow failed\ (x, Mathematics) \land\forall\ (y)\ [\neg(x==y) \land student(y) \rightarrow \neg failed\ (x, Mathematics)].$$

- **PL**
- It uses prepositions in which complete sentence is denoted by symbol.
- Pl Cannot represent individual entities Eg: meera is short.
- It cannot express generalization, specialization or pattern.

  ex : triangles have 3 sides.
- **FOL**
- Fol uses predicates which involve constants, variables, functions relations
- Fol can represent individual properties {short(meera)}
- It can express generalization, specialization or pattern.
- Ex: no_of_sides(triangle,3)

- We use the rules to derive the soundness of the argument in Example.
- Again let the following propositions stand for these statements about the world:
- P: Rani writes books.
- Q: Rani helps other people to write books.
- R: Rani earns her living as an editor

The following derivation determines Rani’s occupation:

	Derivation	Rule
1	$P \vee Q$	Premise
2	$\neg P$	Premise
3	$Q \Rightarrow R$	Premise
4	Q	1, 2, DS
5	R	3, 4, MP

Disjunctive Syllogism: Notation of Disjunctive syllogism: $\dfrac{P \vee Q, \quad \neg P}{Q}$

P: Rani writes books.

Q: Rani helps other people to write books.

Given premise $\neg P$ is true , Q is true

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

- Q: Rani helps other people to write books.
- R: Rani earns her living as an editor

Given premise Q is true, R is true.

Therefore Rani's Occupation is Editor

- Let the following propositions stand for these statements about the world:

- P: Robbery was the reason for the murder.
- Q: Something was taken.
- R: Politics was the reason for the murder.
- S: A woman was the reason for the murder.
- T: The murderer left immediately.
- U: The murderer left tracks all over the room.

- The following derivation determines the reason for the murder

	Derivation	Rule
1	$\neg$Q	Premise
2	P$\Rightarrow$Q	Premise
3	$\neg$P$\Rightarrow$R$\vee$S	Premise
4	R$\Rightarrow$T	Premise
5	U	Premise
6	U$\Rightarrow$ $\neg$T	Premise
7	$\neg$P	1, 2, MT
8	R$\vee$S	3 7, MP
9	$\neg$T	5, 6, MP
10	$\neg$R	4, 9, MT
11	S	8, 10, DS

Inference Rule	Name	
$A, B \vDash A \wedge B$	Combination rule	CR
$A \wedge B \vDash A$	Simplification rule	SR
$A \vDash A \vee B$	Addition rule	AR
$A, A \Rightarrow B \vDash B$	Modus ponens	MP
$\neg B, A \Rightarrow B \vDash \neg A$	Modus tolens	MT
$A \Rightarrow B, B \Rightarrow C \vDash A \Rightarrow C$	Hypothetical syllogism	HS
$A \vee B, \neg A \vDash B$	Disjunctive syllogism	DS
$A \Rightarrow B, \neg A \Rightarrow B \vDash B$	Rule of cases	RC
$A \Leftrightarrow B \vDash A \Rightarrow B$	Equivalence elimination	EE
$A \Rightarrow B, B \Rightarrow A \vDash A \Leftrightarrow B$	Equivalence introduction	EI
$A, \neg A \vDash B$	Inconsistency rule	IR
$A \wedge B \vDash B \wedge A$	"and" Commutivity rule	ACR
$A \vee B \vDash B \vee A$	"or" Commutivity rule	OCR
If $A_1, A_2, \ldots, A_n, B \vDash C$ then $A_1, A_2, \ldots, A_n \vDash B \Rightarrow C$	Deduction theorem	DT

A set of inference rules is called a deduction system.

1. Lucy* is a professor
2. All professors are people.
3. John is the dean.
4. Deans are professors.
5. All professors consider the dean a friend or don’t know him.
6. Everyone is a friend of someone.
7. People only criticize people that are not their friends.
8. Lucy criticized John .

Knowledge base:

- is-prof(lucy)
- $\forall$ x (is-prof(x) $\rightarrow$ is-person(x))
- is-dean(John)
- $\forall$ x (is-dean(x) $\rightarrow$ is-prof(x))
- $\forall$ x ($\forall$ y (is-prof(x) $\wedge$ is-dean(y) $\rightarrow$ is-friend-of(y,x) $\vee$ $\neg$knows(x, y)))
- $\forall$ x ($\exists$ y (is-friend-of (y, x)))
- $\forall$ x ($\forall$ y (is-person(x) $\wedge$ is-person(y) $\wedge$ criticize (x,y) $\rightarrow$ $\neg$is-friend-of (y,x)))
- criticize(lucy, John)

Question: Is John no friend of Lucy?

Inference in First-Order Logic

- Inference in First-Order Logic is used to deduce new facts or sentences from existing sentences. Before understanding the FOL inference rule, let's understand some basic terminologies used in FOL.

Substitution:

- Substitution is a fundamental operation performed on terms and formulas. It occurs in all inference systems in first-order logic. The substitution is complex in the presence of quantifiers in FOL. If we write **F[a/x]**, so it refers to substitute a constant "**a**" in place of variable "**x**".

Equality

- **Equality:**First-Order logic does not only use predicate and terms for making atomic sentences but also uses another way, which is equality in FOL. For this, we can use **equality symbols** which specify that the two terms refer to the same object.
- **Example: Brother (John) = Smith.**

- As in the above example, the object referred by the **Brother (John)** is similar to the object referred by **Smith**. The equality symbol can also be used with negation to represent that two terms are not the same objects.
- **Example: $\neg(x=y)$ which is equivalent to $x \neq y$.**

FOL inference rules for quantifier:

- As propositional logic we also have inference rules in first-order logic, so following are some basic inference rules in FOL:
- **Universal Generalization**
- **Universal Instantiation**
- **Existential Instantiation**
- **Existential introduction**

Universal Generalization:

- Universal generalization is a valid inference rule which states that if premise $P(c)$ is true for any arbitrary element c in the universe of discourse, then we can have a conclusion as $\forall x\, P(x)$.

It can be represented as: $\frac{P(c)}{\forall x\, P(x)}$.

Example: Let's represent, P(c): "**A byte contains 8 bits**", so for **∀ x P(x)** "**All bytes contain 8 bits**.", it will also be true.

Universal Instantiation:

- Universal instantiation is also called as universal elimination or UI is a valid inference rule. It can be applied multiple times to add new sentences.
- The new KB is logically equivalent to the previous KB.
- As per UI, **we can infer any sentence obtained by substituting a ground term for the variable**.

It can be represented as: $\frac{\forall x\, P(x)}{P(c)}$.

Example:1.

IF "Every person like ice-cream"=> ∀x P(x) so we can infer that "John likes ice-cream" => P(c)

Example: 2.

- Let's take a famous example,
- "All kings who are greedy are Evil." So let our knowledge base contains this detail as in the form of FOL:
- **∀x king(x) ∧ greedy (x) → Evil (x),**
- So from this information, we can infer any of the following statements using Universal Instantiation:
- **King(John) ∧ Greedy (John) → Evil (John),**
- **King(Richard) ∧ Greedy (Richard) → Evil (Richard),**
- **King(Father(John)) ∧ Greedy (Father(John)) → Evil (Father(John)),**

Existential Instantiation:

- Existential instantiation is also called as Existential Elimination, It can be applied only once to replace the existential sentence.
- The new KB is not logically equivalent to old KB, but it will be satisfiable if old KB was satisfiable.

It can be represented as: $\dfrac{\exists x\, P(x)}{P(c)}$

Example:

- From the given sentence: **∃x Crown(x) ∧ OnHead(x, John),**
- So we can infer: **Crown(K) ∧ OnHead(K, John),** as long as K does not appear in the knowledge base.
- The above used K is a constant symbol, which is called **Skolem constant**.
- The Existential instantiation is a special case of **Skolemization process**.

Existential Introduction

- An existential introduction is also known as an existential generalization, which is a valid inference rule in first-order logic.
- This rule states that if there is some element c in the universe of discourse which has a property P, then we can infer that there exists something in the universe which has the property P.

It can be represented as: $\frac{P(c)}{\exists x P(x)}$

- **Example:**
- **Let's say that,**
  "Priyanka got good marks in English."
  "Therefore, someone got good marks in English."

Well Formed Formulas

- Not all strings can represent propositions of the predicate logic.
- Those which produce a proposition when their symbols are interpreted must follow the rules given below, and they are called **wffs**(well-formed formulas) of the first order predicate logic.

Rules for constructing Wffs

- A predicate name followed by a list of variables such as $P(x, y)$, where P is a predicate name, and x and y are variables, is called an **atomic formula**.

- **Wffs are constructed using the following rules:**

1. *True* and *False* are wffs.
2. Each propositional constant (i.e. specific proposition), and each propositional variable (i.e. a variable representing propositions) are wffs.
3. Each atomic formula (i.e. a specific predicate with variables) is a wff.

4. If A, B, and C are wffs, then so are $\neg A$, $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, and $(A \leftrightarrow B)$.
5. If x is a variable (representing objects of the universe of discourse), and A is a wff, then so are $\forall x A$ and $\exists x A$.

Example:-

Every person has a father

$\forall x(\text{Person}(x) \rightarrow \text{Father}(x))$

Person(x), x is a person

Father(x), x has father

Example:-

There is a man and he is the father of Ram

$\text{Man}(x) \land \text{Father}(x,\text{Ram})$

Man(x), x is a man

Father(x,Ram), x is the father of Ram

Example:-

All Dancers love to dance

$\forall x(Dancer(x) \rightarrow Love(x,Dance))$

Dancer(x), x is a dancer

Love(x,dance), x love to dance

Example:-

Everone who signs and plays instrument love to dance

$\forall x((Sing(x) \wedge Plays(x)) \rightarrow Love(x,Dance)$

Sing(x), x who sings

Plays(x), x who plays an instrument

Love(x,Dance), x love to dance

Resolution

- Resolution is a theorem proving technique that proceeds by building refutation proofs, i.e., proofs by contradictions.
- Resolution is used, if there are various statements are given, and we need to prove a conclusion of those statements.
- Resolution is a single inference rule which can efficiently operate on the **conjunctive normal form or clausal form**.
- **Conjunctive Normal Form**: A sentence represented as a conjunction of clauses is said to be **conjunctive normal form** or **CNF**.

Steps for Resolution:

- Conversion of facts into first-order logic (FOL).
- Convert FOL statements into conjunctive normal form (CNF).
- Negate the statement which needs to prove (proof by contradiction)
- Draw resolution graph (unification).

Facts to First Order Logic (FOL) in Artificial Intelligence

a. John likes all kind of food.

b. Apple and vegetable are food

c. Anything anyone eats and not killed is food.

d. Anil eats peanuts and still alive

e. Harry eats everything that Anil eats.

f. John likes peanuts.

a. $\forall x: \text{food}(x) \rightarrow \text{likes}(\text{John}, x)$

b. $\text{food}(\text{Apple}) \wedge \text{food}(\text{vegetables})$

c. $\forall x \forall y: \text{eats}(x, y) \wedge \neg \text{killed}(x) \rightarrow \text{food}(y)$

d. $\text{eats}(\text{Anil}, \text{Peanuts}) \wedge \text{alive}(\text{Anil})$.

e. $\forall x : \text{eats}(\text{Anil}, x) \rightarrow \text{eats}(\text{Harry}, x)$

f. $\forall x: \neg \text{killed}(x) \rightarrow \text{alive}(x)$

g. $\forall x: \text{alive}(x) \rightarrow \neg \text{killed}(x)$

added predicates

h. $\text{likes}(\text{John}, \text{Peanuts})$

Convert FOL to CNF

- Eliminate all implications($\rightarrow$) and rewrite ; if a$\rightarrow$b then $\neg \mathbf{a} \vee \mathbf{b}$
- Move negation ($\neg$) inwards and rewrite
- Rename the variables or standardize variables
- Drop the universal quantifiers

- **Eliminate all implication (→) and rewrite**

a. $\forall x$: food(x) → likes(John, x)

b. food(Apple) ∧ food(vegetables)

c. $\forall x$ $\forall y$: eats(x, y) ∧ ¬ killed(x) → food(y)

d. eats (Anil, Peanuts) ∧ alive(Anil).

e. $\forall x$: eats(Anil, x) → eats(Harry, x)

f. $\forall x$: ¬ killed(x) → alive(x)

g. $\forall x$: alive(x) →¬ killed(x)

h. likes(John, Peanuts)

a. $\forall x$ ¬ food(x) V likes(John, x)

b. food(Apple) Λ food(vegetables)

c. $\forall x$ $\forall y$ ¬ [eats(x, y) Λ ¬ killed(x)] V food(y)

d. eats (Anil, Peanuts) Λ alive(Anil)

e. $\forall x$ ¬ eats(Anil, x) V eats(Harry, x)

f. $\forall x$¬ [¬ killed(x)] V alive(x)

g. $\forall x$ ¬ alive(x) V ¬ killed(x)

h. likes(John, Peanuts).

- **Move negation (¬)inwards and rewrite**

a. $\forall x \neg food(x) \lor likes(John, x)$

b. $food(Apple) \land food(vegetables)$

c. $\forall x \forall y \neg [eats(x, y) \land \neg killed(x)] \lor food(y)$

d. $eats\ (Anil, Peanuts) \land alive(Anil)$

e. $\forall x \neg eats(Anil, x) \lor eats(Harry, x)$

f. $\forall x \neg [\neg killed(x)] \lor alive(x)$

g. $\forall x \neg alive(x) \lor \neg killed(x)$

h. $likes(John, Peanuts).$

a. $\forall x \neg food(x) \lor likes(John, x)$

b. $food(Apple) \land food(vegetables)$

c. $\forall x \forall y \neg eats(x, y) \lor killed(x) \lor food(y$

d. $eats\ (Anil, Peanuts) \land alive(Anil)$

e. $\forall x \neg eats(Anil, x) \lor eats(Harry, x)$

f. $\forall x \quad killed(x)] \lor alive(x)$

g. $\forall x \neg alive(x) \lor \neg killed(x)$

h. $likes(John, Peanuts).$

- Rename the variables or standardize variables

a. $\forall x \neg food(x) \lor likes(John, x)$

b. $food(Apple) \land food(vegetables)$

c. $\forall x \forall y \neg eats(x, y) \lor killed(x) \lor food(y)$

d. $eats\ (Anil, Peanuts) \land alive(Anil)$

e. $\forall x \neg eats(Anil, x) \lor eats(Harry, x)$

f. $\forall x\ killed(x)\ ] \lor alive(x)$

g. $\forall x \neg alive(x) \lor \neg killed(x)$

h. $likes(John, Peanuts)$.

a. $\forall x \neg food(x) \lor likes(John, x)$

b. $food(Apple) \land food(vegetables)$

c. $\forall y \forall z \neg eats(y, z) \lor killed(y) \lor food(z)$

d. $eats\ (Anil, Peanuts) \land alive(Anil)$

e. $\forall w \neg eats(Anil, w) \lor eats(Harry, w)$

f. $\forall g\ killed(g)\ ] \lor alive(g)$

g. $\forall k \neg alive(k) \lor \neg killed(k)$

h. $likes(John, Peanuts)$.

- Drop the universal quantifiers

a. ∀x ¬ food(x) V likes(John, x)

b. food(Apple) Λ food(vegetables)

c. ∀y ∀z ¬ eats(y, z) V killed(y) V food(z)

d. eats (Anil, Peanuts) Λ alive(Anil)

e. ∀w¬ eats(Anil, w) V eats(Harry, w)

f. ∀g killed(g)] V alive(g)

g. ∀k ¬ alive(k) V ¬ killed(k)

h. likes(John, Peanuts).

a. ¬ food(x) V likes(John, x)

b. food(Apple)

c. food(vegetables)

d. ¬ eats(y, z) V killed(y) V food(z)

e. eats (Anil, Peanuts)

f. alive(Anil)

g. ¬ eats(Anil, w) V eats(Harry, w)

h. killed(g) V alive(g)

i. ¬ alive(k) V ¬ killed(k)

j. likes(John, Peanuts).

Draw Resolution Tree in Artificial Intelligence

a. ¬ food(x) V likes(John, x)

b. food(Apple)

c. food(vegetables)

d. ¬ eats(y, z) V killed(y) V food(z)

e. eats (Anil, Peanuts)

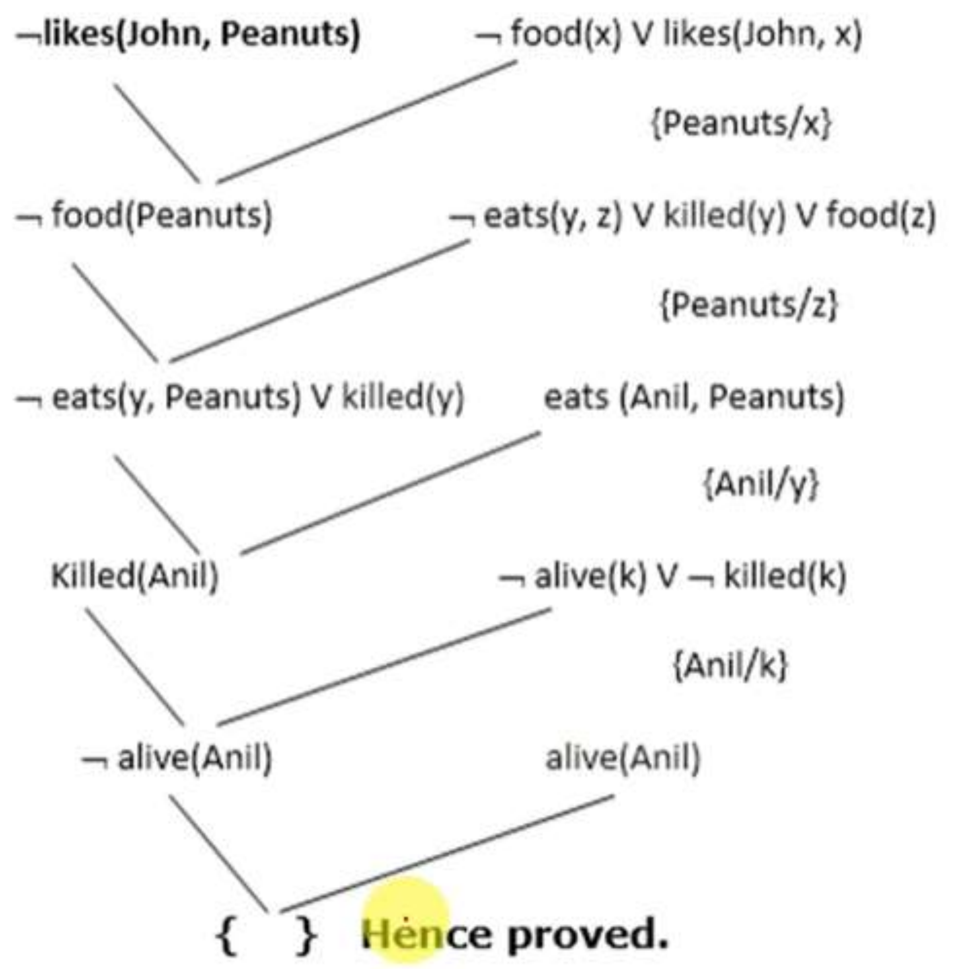
f. alive(Anil)

g. ¬ eats(Anil, w) V eats(Harry, w)

h. killed(g) V alive(g)

i. ¬ alive(k) V ¬ killed(k)

j. likes(John, Peanuts).



- Example:

a) Ravi likes all kind of food.
b) Apple and chicken are food.
c) Anything anyone eats and is not killed is food.
d) Ajay eats peanuts and still alive.

Prove: Ravi likes peanuts.

write in FOL form

- likes(Ravi, Peanuts) => ¬ likes(Ravi, Peanuts)

- Steps to solve Resolution

1. Negate the statement to be proved.
2. Convert given facts into FOL
3. Convert FOL into CNF
4. Draw resolution graph

Convert the facts into FOL

1. $\forall x$: food(x) -> likes(Ravi,x)
2. i) food(apple)

   ii) food(chicken)

3. ∀x∀y: eats(x,y) ∧ ¬killed(x) -> food(y)

4. eats(ajay,peanuts) ∧ alive(ajay)
5. ∀x: ¬killed(x) -> alive(x)

added predicates

6. ∀x: alive(x) -> ¬killed(x)

- Convert FOL into CNF

1. Eliminate ‘->’ & ‘<->’

a->b : ¬a V b

a<->b : a->b ∧ b->a

2. Move ¬ inward

- $\neg(\forall x p) = \exists x \neg p$
- $\neg(\exists x p) = \forall x \neg p$
- $\neg(a \vee b) = \neg a \wedge \neg b$
- $\neg(a \wedge b) = \neg a \vee \neg b$
- $\neg(\neg a) = a$

3. Rename Variable.
4. Replace Existential Quantifier by skolem constant

$$\exists x \, Rich(x) = Rich(G1)$$

5. Drop Universal Quantifier

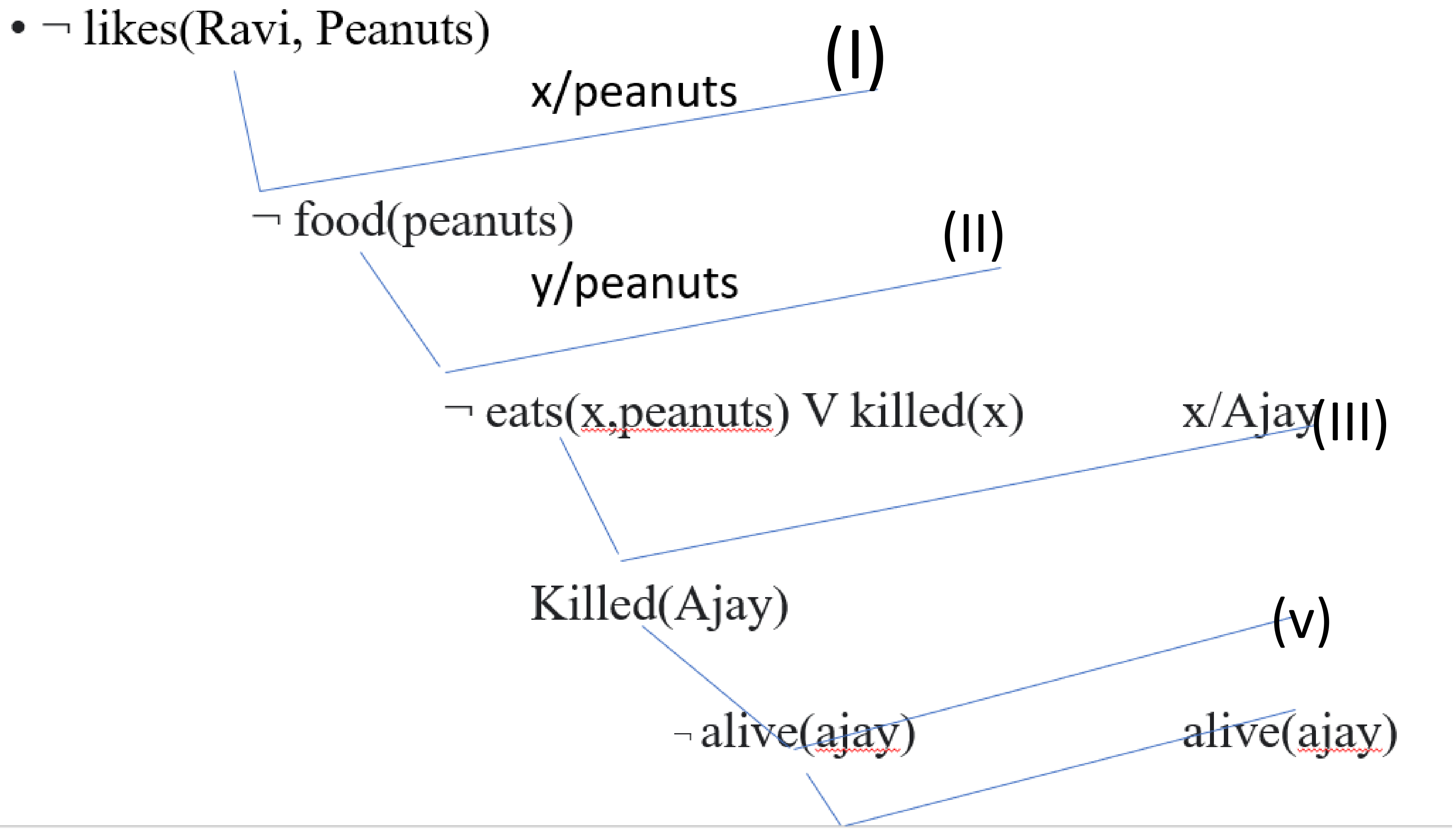
1. ¬ food(x) V likes(Ravi,x) [a->b = ¬a V b]

3. ¬ [eats(x,y) ∧ ¬killed(x)] V food(y)

   - ¬ eats(x,y) V killed(x) V food(y)

4. eats(ajay,peanuts) ∧ alive(ajay) (Divide in resolution graph)
5. ¬ ¬killed(x) V alive(x) : killed(x) V alive(x)
6. ¬ alive(x) -> ¬ ¬ killed(x) : ¬ alive(x) V ¬ killed(x)

Resolution Graph



Uncertain Knowledge and Reasoning

- In real life, it is not always possible to determine the state of the environment as it might not be clear. Due to partially observable or non-deterministic environments, agents may need to handle uncertainty and deal with:
- Uncertain data: Data that is missing, unreliable, inconsistent or noisy
- Uncertain knowledge: When the available knowledge has multiple causes leading to multiple effects or incomplete knowledge of causality in the domain
- Uncertain knowledge representation: The representations which provides a restricted model of the real system, or has limited expressiveness

- Inference: In case of incomplete or default reasoning methods, conclusions drawn might not be completely accurate. Let’s understand this better with the help of an example.

- IF primary infection is bacteriacea
- AND site of infection is sterile
- AND entry point is gastrointestinal tract
- THEN organism is bacteriod (0.7)

- In such uncertain situations, the agent does not guarantee a solution but acts on its own assumptions and probabilities and gives some degree of belief that it will reach the required solution.

- Such uncertain situations can be dealt with using

1. Probability theory
2. Truth Maintenance systems
3. Fuzzy logic.

Probability

- Probability is the degree of likeliness that an event will occur.
- It provides a certain degree of belief in case of uncertain situations.
- It is defined over a set of events U and assigns value P(e) i.e. probability of occurrence of event e in the range [0,1].
- Here each sentence is labeled with a real number in the range of 0 to 1, 0 means the sentence is false and 1 means it is true.

- (For dependent events) Conditional Probability or Posterior Probability is the probability of event A given that B has already occurred.

$$P(A|B) = (P(B|A) * P(A)) / P(B)$$

- For example, P(It will rain tomorrow| It is raining today) represents conditional probability of it raining tomorrow as it is raining today.

NOTE:- $P(A|B) + P(NOT(A)|B) = 1$

- (For independent events)Joint probability is the probability of 2 independent events happening simultaneously like rolling two dice or tossing two coins together.

$$P(A \cap B) = P(A) \times P(B)$$

Bayes Theorem

- Bayes' theorem is also known as **Bayes' rule, Bayes' law**, or **Bayesian reasoning**, which determines the probability of an event with uncertain knowledge.

- In probability theory, it relates the conditional probability and marginal probabilities of two random events.

- It is a way to calculate the value of P(B|A) with the knowledge of P(A|B).

Bayes Theorem: { Reverse Probability}

- Given that red ball is drawn what is the probability that the ball is from bag A.

- $P(A/R) = P(A \wedge R) /((P(A \wedge R) + P(B \wedge R))$

$P(A \wedge R)$: from Bag A

$((P(A \wedge R) + P(B \wedge R))$:Total probability of Red Ball.

=> **$P(A).P(R/A) / P(R)$**

The general statement of Bayes' theorem is

"The conditional probability of an event A,
given the occurrence of another event B,
is equal to the product of the event of B,
given A and the probability of A divided by the probability of event B."

where, **i.e.P(A|B) = P(B|A)P(A) / P(B)**

- P(A) and P(B) are the probabilities of events A and B
- P(A|B) is the probability of event A when event B happens
- P(B|A) is the probability of event B when A happens

- P(A|B) is known as **posterior**, which we need to calculate, and it will be read as Probability of hypothesis A when we have occurred an evidence B.

- P(B|A) is called the likelihood, in which we consider that hypothesis is true, then we calculate the probability of evidence.

- P(A) is called the **prior probability**, probability of hypothesis before considering the evidence.

- P(B) is called **marginal probability**, pure probability of an evidence.

$$P(\text{cause} \mid \text{effect}) = \frac{P(\text{effect} \mid \text{cause})\, P(\text{cause})}{P(\text{effect})}$$

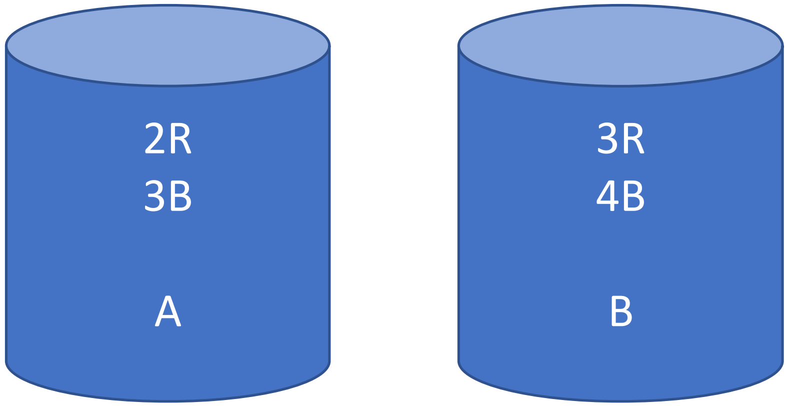
- **Question: what is the probability that a patient has diseases meningitis with a stiff neck?**

Given Data:

- A doctor is aware that disease meningitis causes a patient to have a stiff neck, and it occurs 80% of the time. He is also aware of some more facts, which are given as follows:
- The Known probability that a patient has meningitis disease is 1/30,000.
- The Known probability that a patient has a stiff neck is 2%.
- Let a be the proposition that patient has stiff neck and b be the proposition that patient has meningitis. , so we can calculate the following as:
- $P(a|b) = 0.8$
- $P(b) = 1/30000$
- $P(a)= .02$

$$P(b|a) = \frac{P(a|b)P(b)}{P(a)} = \frac{0.8*(\frac{1}{30000})}{0.02} = 0.001333333.$$

Hence, we can assume that 1 patient out of 750 patients has meningitis disease with a stiff neck.



1. What is the probability of red ball given that bag A is chosen

- P(R/A) = 2/5 { Bag A is Chosen}

2. What is the probability that redball is drawn from bag A.

{ Here we need to select Red ball and Bag A – Two Conditions}

$$P(A \land R) = P(A).\ P(R/A)$$

3. What is the probability of red ball

$$P(R) = P(A \wedge R) + P(B \wedge R)$$

$P(A \wedge R)$: Probability of getting a red ball from bag A

$P(B \wedge R)$: Probability of getting a red ball from bag B

Baysian Belief Network:-

Bayesian belief networks (BBNs) are probabilistic graphical models that are used to represent uncertain knowledge and make decisions based on that knowledge. They are a type of Bayesian network, which is a graphical model that represents probabilistic relationships between variables.

It has 2 elements

1.DAG, is a graphical representation of the variables in the network and the relationships between them

2.Probability Table

Eample:-

In a house a boy, cat and dog are staying

When rain, dog barks(not always)

When dog barks, cat hides(not always)

Hint:- Not always means uncertainity.

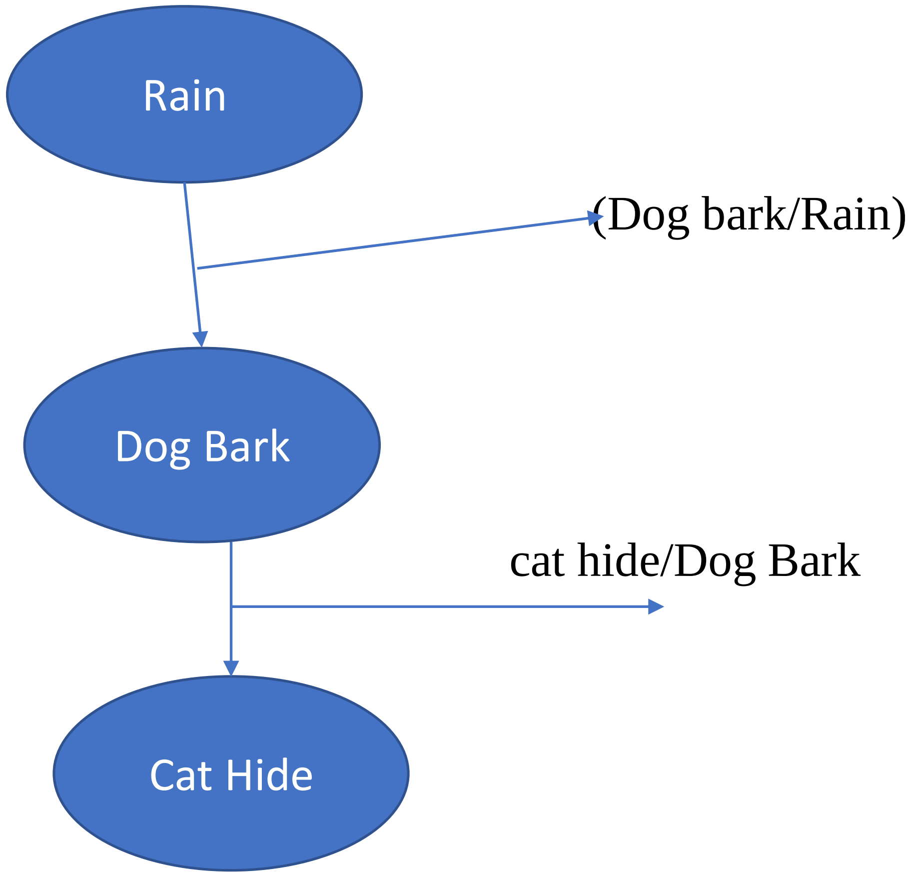
In DAG Directed Arrow represents conditional probability

- Dog bark/Rain: Dog will batk if it rains
- Cat hide/dog bark: Cat will hide if dog barks

In Conditional Probability Table:

- P(Dog/Rain): Probability of considered node with respect to its parent node.
  - P(X/P)

Directed Acyclic Graph (DAG)



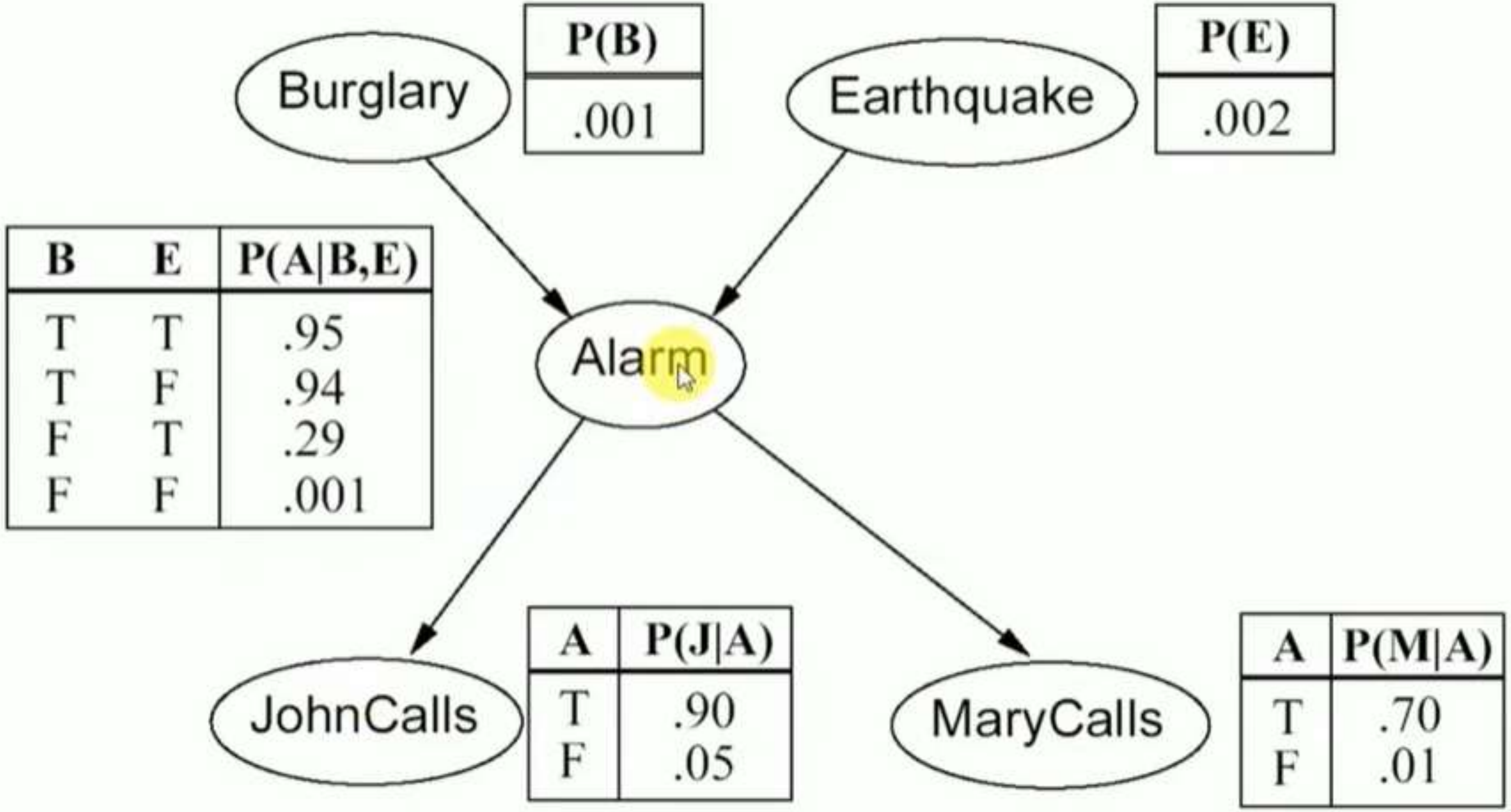
Probability Table

	R	~R
B	9/48	18/48
~B	3/48	18/48

BAYESIAN BELIEF NETWORKS – EXAMPLE – 1

- You have a new burglar alarm installed at home.
- It is fairly reliable at detecting burglary, but also sometimes responds to minor earthquakes.
- You have two neighbors, John and Merry , who promised to call you at work when they hear the alarm.
- John always calls when he hears the alarm, but sometimes confuses telephone ringing with the alarm and calls too.
- Merry likes loud music and sometimes misses the alarm.
- Given the evidence of who has or has not called, we would like to estimate the probability of a burglary.

BAYESIAN BELIEF NETWORKS – EXAMPLE – 1



P(B)
.001

P(E)
.002

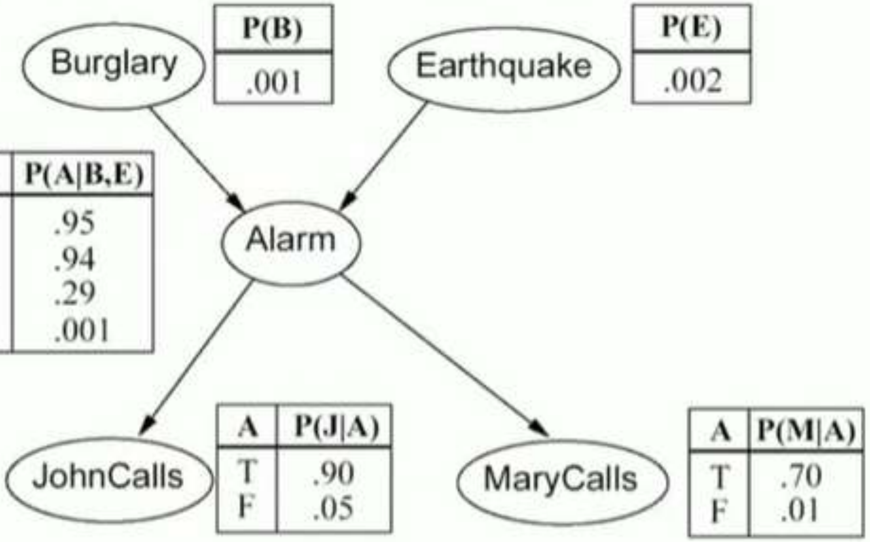
B	E	P(A|B,E)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

A	P(J|A)
T	.90
F	.05

A	P(M|A)
T	.70
F	.01

BAYESIAN BELIEF NETWORKS – EXAMPLE – 1

1. What is the probability that the alarm has sounded but neither a burglary nor an earthquake has occurred, and both John and Merry call?



B	E	P(A|B,E)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

A	P(J|A)
T	.90
F	.05

A	P(M|A)
T	.70
F	.01

Solution:

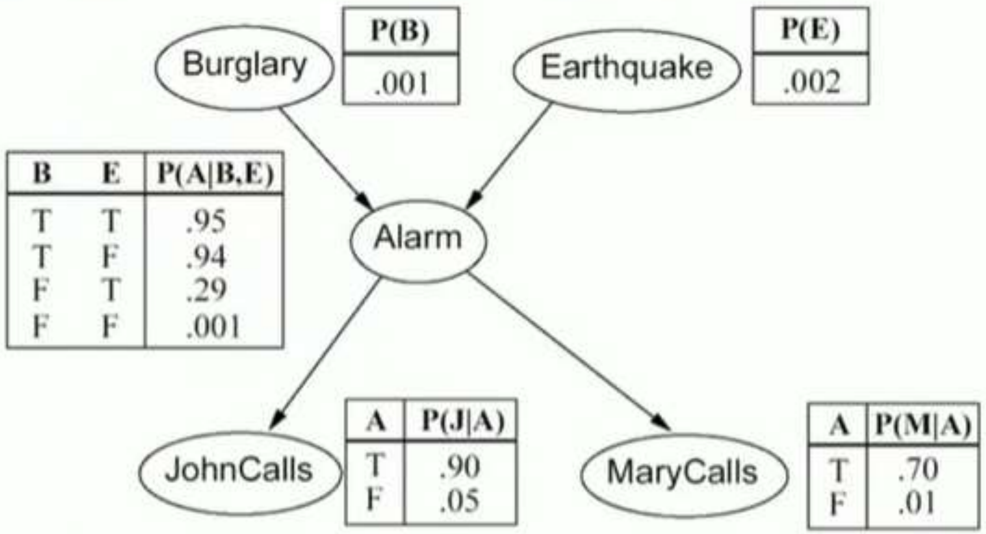
$$P(j \wedge m \wedge a \wedge \neg b \wedge \neg e) = P(j \mid a)\, P(m \mid a)\, P(a \mid \neg b, \neg e)\, P(\neg b)\, P(\neg e)$$

$$= 0.90 \times 0.70 \times 0.001 \times 0.999 \times 0.998$$

$$= 0.00062$$

BAYESIAN BELIEF NETWORKS – EXAMPLE – 1

2. What is the probability that John call?



Solution:

$P(j) = P(j \mid a)\,P(a) + P(j \mid \neg a)\,P(\neg a)$

$= P(j|a)\{P(a|b,e)*P(b,e)+P(a|\neg b,e)*P(\neg b,e)+P(a|b,\neg e)*P(b,\neg e)+P(a|\neg b,\neg e)*P(\neg b,\neg e)\}$

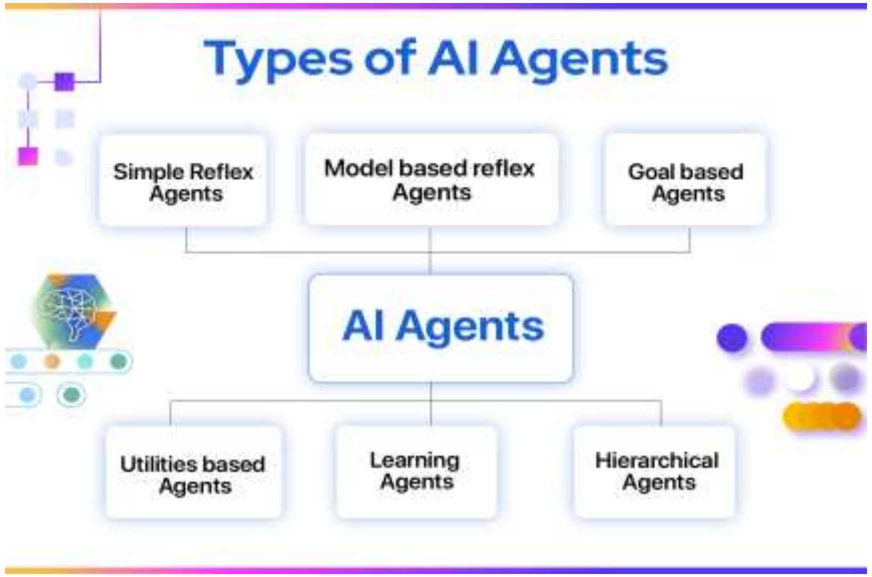
$+ P(j|\neg a)\{P(\neg a|b,e)*P(b,e)+P(\neg a|\neg b,e)* P(\neg b,e)+P(\neg a|b,\neg e)* P(b,\neg e)+P(\neg a|\neg b, \neg e)*$

$P(\neg b, \neg e)\}$

$= 0.90 * 0.00252 + 0.05 * 0.9974 = 0.0521$

AI Agents

In artificial intelligence, an agent is a computer program or system that is designed to perceive its environment, make decisions and take actions to achieve a specific goal or set of goals. The agent operates autonomously, meaning it is not directly controlled by a human operator.



Utility-Based System

Utility based systems are used to make rational decisions based on a utility function.

Rational Decision making means picking the option that maximizes an agent's expected utility. i.e. give the best outcome.

Utility-based agents are a type of intelligent agent in artificial intelligence (AI) that make decisions based on a utility function. This function measures the degree of satisfaction or utility associated with different possible outcomes.

Decision Network

Decision networks are graphical models used to represent and solve decision-making problems.

They extend Bayesian networks by incorporating decision and utility nodes, allowing for a comprehensive analysis of decision scenarios.

A decision network consists of three types of nodes:

- **Chance Nodes**: random variables and their possible values, capturing the uncertainty in the decision-making process.
- **Decision Node**: the choices available to the decision-maker.
- **Utility Node**: utility or value of the outcome to evaluate and compare different decision paths.
- NOTE:- The structure of a decision network is typically represented as a directed acyclic graph (DAG), where **Arcs (Edges)** Indicate relationships between nodes.