Knowledge Representation

Unit 9.3

Knowledge

Knowledge is a theoretical or practical understanding of a subject or a domain.

Knowledge is also the sum of what is currently known.

Types of knowledge:

Classification-based Knowledge :Ability to classify information

Decision-oriented Knowledge: Choosing the best option

Descriptive knowledge: State of some world (heuristic)

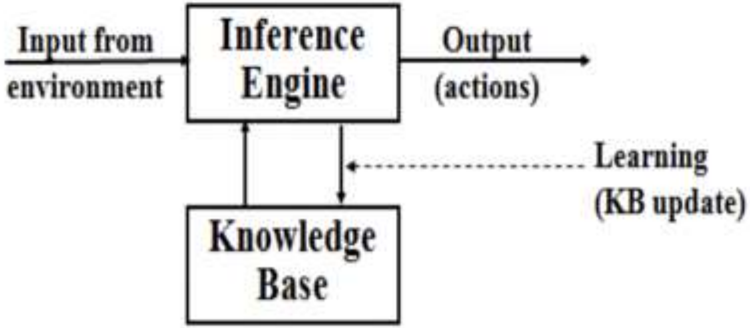
Procedural knowledge: How to do something

Reasoning knowledge: What conclusion is valid in what situation?

Assimilative knowledge: What its impact is?

A Knowledge-Based Agent

- A knowledge-based agent consists of a knowledge base (KB) and an inference engine (IE).



- A knowledge-base is a set of sentences of what one knows about the world.
- The Inference engine derives new sentences from the input and KB.
- The agent operates as follows:
  1. It receives percepts from environment
  2. It computes what action it should perform (by IE and KB)
  3. It performs the chosen action.

The objective of knowledge representation is to express the knowledge about the world in a computer-tractable form.

This computer- tractable form of knowledge helps the agent to identify patterns of good reasoning and patterns of bad reasoning, so the agent know which to follow and which to avoid.

A formal language is required to represent knowledge in a computer tractable form and reasoning processes are required to manipulate this knowledge to deduce new facts.

Key aspects of knowledge representation languages are: –

- Syntax: describes how sentences are formed in the language.
- Semantics: describes the meaning of sentences, what is it the sentence refers to in the real world.
- Proof Theory(Inference method): Set of rules for generating new sentences that are necessarily true given that the old sentences are true

Knowledge Representation using Logic: –

Logic is defined as a formal language for expressing knowledge and ways of reasoning.

Therefore, it should have syntax, semantics and inference method.

Syntax: describes how sentences are formed in the LOGIC.

Semantics: describes the meaning of sentences.

Inference method: set of rules for generating new sentences.

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.

So, Logic is:

Concise, unambiguous, context insensitive, expressive, effective for inferences

Examples of Logics are:

Propositional logic

Predicate Logic and

Fuzzy Logic

Propositional Logic

Propositional logic is the simplest formal logic for the representation of the knowledge in terms of propositions.

Proposition is a declarative statement that is either true or false but not both.

If a proposition is true, then we say it has a truth value of "true"; if a proposition is false, its truth value is "false".

Some examples of Propositions are given below :

"Man is Mortal", it returns truth value —TRUE‖

"12 + 9 = 3 – 2", it returns truth value —FALSE‖

The following sentences are not Proposition:

"A is less than 2". It is because unless we give a specific value of A, we cannot say whether the statement is true or false.

Also the sentences "Close the door", and "Is it hot outside ?"are not propositions.

Syntax of Propositional Logic

Syntax of the propositional logic defines the:

- Which symbols can be use (English: letters, punctuation)
- Rules for constructing legal sentences in the logic.
- How we are allowed to combine symbols

Symbols:

- Logical constants: true, false
- Propositional symbols: P, Q, R, S, ... , etc
- Wrapping parentheses: (…)

- Atomic formulas(Sentence): Propositional Symbols or logical constants.
- Literals: atomic sentences and their negations
- Complex Formulas: can be formed by combining atomic formulas with the following connectives:

$\neg$ ...not	[negation]
$\wedge$ ...and	[conjunction]
$\vee$ ...or	[disjunction]
$\longrightarrow$...implies	[implication / conditional]
$\longleftrightarrow$..is equivalent	[biconditional]

- A sentence 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 above rules

- **Order of precedence** of logical connectors

¬ highest precedence

^

v

=>

<=> lowest precedence

e.g. $\neg P \vee Q \wedge R \Rightarrow S$ is equivalent to $((\neg P) \vee (Q \wedge R)) \Rightarrow S$

Examples of PL sentences

- P means "It is hot"
- Q means "It is humid"
- R means "It is raining"
- P ^ Q => R

  "If it is hot and humid, then it is raining"
- Q => P

  "If it is humid, then it is hot"
- Q

  "It is humid."

- How to compute the truth of sentences formed with each of the five connectives?
  - For complex sentences, we have five rules, which hold for any sub-sentences P and Q in any model m:

    $\neg P$ is true iff P is false in m.
    $P \wedge Q$ is true iff both P and Q are true in m.
    $P \vee Q$ is true iff either P or Q is true in m.
    $P \Rightarrow Q$ is true unless P is true and Q is false in m.
    $P \Leftrightarrow Q$ is true iff P and Q are both true or both false in m.

  - The above rules can be summarized as follows:

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*

Properties

- Validity(Tautology)
- Satisfiability (contingency)
- Un-Satisfiability (Contradictory)
- Equivalent
- Entailment
- Completeness
- Soundness

- Validity:
  - A sentence is valid if it is true in all models,

    e.g., *True*, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

  - Valid sentences are also known as tautologies. Every valid sentence is logically equivalent to True

- **Example** : Prove $[(A \rightarrow B) \wedge A] \rightarrow B$ is a tautology

  Solution: The truth table is as follows

A	B	$A \rightarrow B$	$(A \rightarrow B) \wedge A$	$[(A \rightarrow B) \wedge A] \rightarrow B$
True	True	True	True	True
True	False	False	False	True
False	True	True	False	True
False	False	True	False	True

As we can see every value of $[(A \rightarrow B) \wedge A] \rightarrow B$ is "True", it is a tautology.

- Satisfiability:
  - A sentence is satisfiable if it is true in some model
    - E.g., A V A
  - Satisfiable sentences are also known as Contingency.

- **Example:** Prove (A∨B)∧(¬A)a contingency

Solution: The truth table is as follows

A	B	A ∨ B	¬ A	(A ∨ B) ∧ (¬ A)
True	True	True	False	False
True	False	True	False	False
False	True	True	True	True
False	False	False	True	False

As we can see every value of (A∨B)∧(¬A) has both "True" and "False", it is a contingency.

- Un-Satisfiability (Contradictory) :
  - A sentence is un-satisfiable if it is true in no models
    - E.g., $A \wedge \neg A$
  - Un-Satisfiable sentences are also known as contradictory sentences.
- Example: Prove $(A \vee B) \wedge [(\neg A) \wedge (\neg B)]$ is a contradiction

  Solution: The truth table is as follows :

A	B	A ∨ B	¬ A	¬ B	(¬ A) ∧ (¬ B)	(A ∨ B) ∧ [(¬ A) ∧ (¬ B)]
True	True	True	False	False	False	False
True	False	True	False	True	False	False
False	True	True	True	False	False	False
False	False	False	True	True	True	False

As we can see every value of $(A \vee B) \wedge [(\neg A) \wedge (\neg B)]$ is "False", it is a contradiction.

- Propositional Equivalences
- Two statements X and Y are logically equivalent if any of the following two conditions hold :
  - The truth tables of each statement have the same truth values.
  - The bi-conditional statement $X \Leftrightarrow Y$ is a tautology.
- **Example** : Prove $\neg(A \vee B)$ and $[(\neg A) \wedge (\neg B)]$ are equivalent

  Testing by 1st method (Matching truth table)

A	B	A ∨ B	¬ (A ∨ B)	¬ A	¬ B	[(¬ A) ∧ (¬ B)]
True	True	True	False	False	False	False
True	False	True	False	False	True	False
False	True	True	False	True	False	False
False	False	False	True	True	True	True

Here, we can see the truth values of $\neg(A \vee B)$ and $[(\neg A) \wedge (\neg B)]$ are same, hence the statements are equivalent.

- Testing by 2nd method (Bi-conditionality)

A	B	$\neg (A \vee B)$	$[(\neg A) \wedge (\neg B)]$	$[\neg (A \vee B)] \Leftrightarrow [(\neg A) \wedge (\neg B)]$
True	True	False	False	True
True	False	False	False	True
False	True	False	False	True
False	False	True	True	True

- As $[\neg(A \vee B)] \Leftrightarrow [(\neg A) \wedge (\neg B)]$ is a tautology, the statements are equivalent.

- Example2:
  - Symbols:
    - P is "It is hot",
    - Q is "It is humid" and
    - R is "It is raining".
  - KB:
    - P^Q=>R ("If it is hot and humid, then it is raining"),
    - Q=>P ("If it is humid, then it is hot"),
    - Q ("It is humid").
  - Question:
    - Is it raining? (i.e., is R entailed by KB?)

P, Q, R	$P \wedge Q \Rightarrow R$	$Q \Rightarrow P$	Q	KB	R	$KB \Rightarrow R$
T, T, T	T	T	T	T	T	T
T, T, F	F	T	T	F	F	T
T, F, T	T	T	F	F	T	T
T, F, F	T	T	F	F	F	T
F, T, T	T	F	T	F	T	T
F, T, F	T	F	T	F	F	T
F, F, T	T	T	F	F	T	T
F, F, F	T	T	F	F	F	T

  - So, R is entailed by the KB and we can conclude it is raining.

Deductive (proof) statements

Done by applying rules of inference directly to the sentences in our knowledge base to construct a proof of the desired sentence without consulting models.

– Example:

- All men are mortal.
- Ram is a man.
- Therefore,Ram is mortal.

If the number of models is large but the length of the proof is short, then Deductive proof can be more efficient than model checking.

Important Logical Equivalences

$\sim(\sim p) \equiv p$	**the Double Negative Law**
$\sim(p \vee q) \equiv (\sim p) \wedge (\sim q)$ $\sim(p \wedge q) \equiv (\sim p) \vee (\sim q)$	**De Morgan's Laws**
$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$ $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$	**the Distributive Laws**

Rule	Tautology	Name
$\begin{array}{l} p \to q \\ p \\ \hline \therefore q \end{array}$	$((p \to q) \wedge p) \Rightarrow q$	Modus Ponens (Law of Detachment)
$\begin{array}{l} p \to q \\ \neg q \\ \hline \therefore \neg p \end{array}$	$((p \to q) \wedge \neg q) \Rightarrow \neg q$	Modus Tollens
$\begin{array}{l} p \to q \\ q \to r \\ \hline \therefore p \to r \end{array}$	$((p \to q) \wedge (q \to r)) \Rightarrow (p \to r)$	Hypothetical Syllogism (Transitivity)
$\begin{array}{l} p \vee q \\ \neg p \\ \hline \therefore q \end{array}$	$((p \vee q) \wedge \neg p) \Rightarrow q$	Disjunctive Syllogism
$\begin{array}{l} p \\ \hline \therefore p \vee q \end{array}$	$p \Rightarrow p \vee q$	Addition
$\begin{array}{l} p \wedge q \\ \hline \therefore p \end{array}$	$(p \wedge q) \Rightarrow p$	Simplification
$\begin{array}{l} p \\ q \\ \hline \therefore p \wedge q \end{array}$	$(p) \wedge (q) \Rightarrow (p \wedge q)$	Conjunction
$\begin{array}{l} p \vee q \\ \neg p \vee r \\ \hline \therefore q \vee r \end{array}$	$((p \vee q) \wedge (\neg p \vee r)) \Rightarrow (q \vee r)$	Resolution

Let’s look at an example for each of these rules to help us make sense of things.

Let p be “It is raining,” and q be “I will make tea,” and r be “I will read a book.”

Example: Modus Ponens

“If it is raining, then I will make tea.”	$p \rightarrow q$	**Modus Ponens (Law of Detachment)**
“It is raining.”	p	$\left(\left(p \rightarrow q\right) \wedge p\right) \Rightarrow q$
“Therefore, I will make tea.”	$\therefore q$	

Some Equivalence Laws

Idempotency	$P \vee P = P$ $P \& P = P$
Associativity	$(P \vee Q) \vee R = P \vee (Q \vee R)$ $(P \& Q) \& R = P \& (Q \& R)$
Commutativity	$P \vee Q = Q \vee P$ $P \& Q = Q \& P$ $P \leftrightarrow Q = Q \leftrightarrow P$
Distributivity	$P \& (Q \vee R) = (P \& Q) \vee (P \& R)$ $P \vee (Q \& R) = (P \vee Q) \& (P \vee R)$
De Morgan's laws	$\sim(P \vee Q) = \sim P \& \sim Q$ $\sim(P \& Q) = \sim P \vee \sim Q$
Conditional elimination	$P \rightarrow Q = \sim P \vee Q$
Bi-conditional elimination	$P \leftrightarrow Q = (P \rightarrow Q) \& (Q \rightarrow P)$

Semantics for Propositional logic

The semantics or meaning of a sentence is just the value true or false. The terms used for semantics of a language are given below.

Valid	A sentence is valid if it is true for all interpretations. Valid sentences are also called tautologies.
Model	An interpretation of a formula or sentence under which the formula is true is called a model of that formula.
Unsatisfiable (contradiction)	It is said to be unsatisfiable if it is false for every interpretation.
Satisfiable	It is said to be satisfiable if it is true for some interpretation.
Equivalence	Two sentences are equivalent if they have the same truth value under every interpretation.

Two Normal (Canonical) Forms

All wffs can be expressed in the following to normal forms

1. CNF (Conjunctive Normal Form)

e.g.: $\underbrace{(A \vee \neg B)}_{\text{Clause 1}} \wedge \underbrace{(B \vee \neg D \vee \neg C)}_{\text{clause 2}}$

2. DNF (Disjunctive Normal Form)

e.g.: $\underbrace{(A \wedge \neg B)}_{\text{models}} \vee \underbrace{(B \wedge \neg D \wedge \neg C)}_{\text{models}}$

Conversion to CNF

- A sentence that is expressed as a conjunction of disjunctions of literals is said to be in conjunctive normal form (CNF). A sentence in CNF that contains only k literals per clause is said to be in k-CNF.
- *Algorithm:*
  - Eliminate $\leftrightarrow$ rewriting $P\leftrightarrow Q$ as $(P\rightarrow Q)\wedge(Q\rightarrow P)$
  - Eliminate $\rightarrow$ rewriting $P\rightarrow Q$ as $\neg P\vee Q$
  - Use De Morgan's laws to push $\neg$ inwards:
    - rewrite $\neg(P\wedge Q)$ as $\neg P\vee\neg Q$
    - rewrite $\neg(P\vee Q)$ as $\neg P\wedge\neg Q$
  - Eliminate double negations: rewrite $\neg\neg P$ *as* P
  - Use the distributive laws to get CNF:
    - rewrite $(P\wedge Q)\vee R$ as $(P\vee R)\wedge(Q\vee R)$
  - Flatten nested clauses:
    - $(P\wedge Q) \wedge R$ as $P\wedge Q \wedge R$
    - $(P\vee Q)\vee R$ as $P\vee Q\vee R$

Proof by resolution

- Resolution is used with knowledge bases in CNF (or clausal form), and is complete for propositional logic.

- Resolution takes two clauses and produces a new clause containing all the literals of the two original clauses except the two complementary literals.

- In addition the resulting clause should contain only one copy of each literal. The removal of multiple copies of literals is called **factoring.**

- For example, if we resolve $(A \vee B)$ with $(A \vee \neg B)$, we obtain $(A \vee A)$, which is reduced to just A.

- **Resolution Algorithm:**

  - Convert KB into CNF
  - Add negation of sentence to be entailed into KB i.e. $(KB \wedge \neg\alpha)$
  - Then apply resolution rule to resulting clauses.
  - The process continues until:
    - There are no new clauses that can be added

      Hence *KB* **does not** entail α
    - Two clauses resolve to entail the empty clause.

      Hence *KB* **does** entail α

Resolution Example1

Example: Consider the knowledge base given as: $KB = (B \Leftrightarrow (A \vee C)) \wedge \neg B$
Prove that $\neg A$ can be inferred from above KB by using resolution.

<u>Solution:</u>

At first, convert KB into CNF

$B \Rightarrow (A \vee C)) \wedge ((A \vee C) \Rightarrow B) \wedge \neg B$

$(\neg B \vee A \vee C) \wedge (\neg(A \vee C) \vee B) \wedge \neg B$

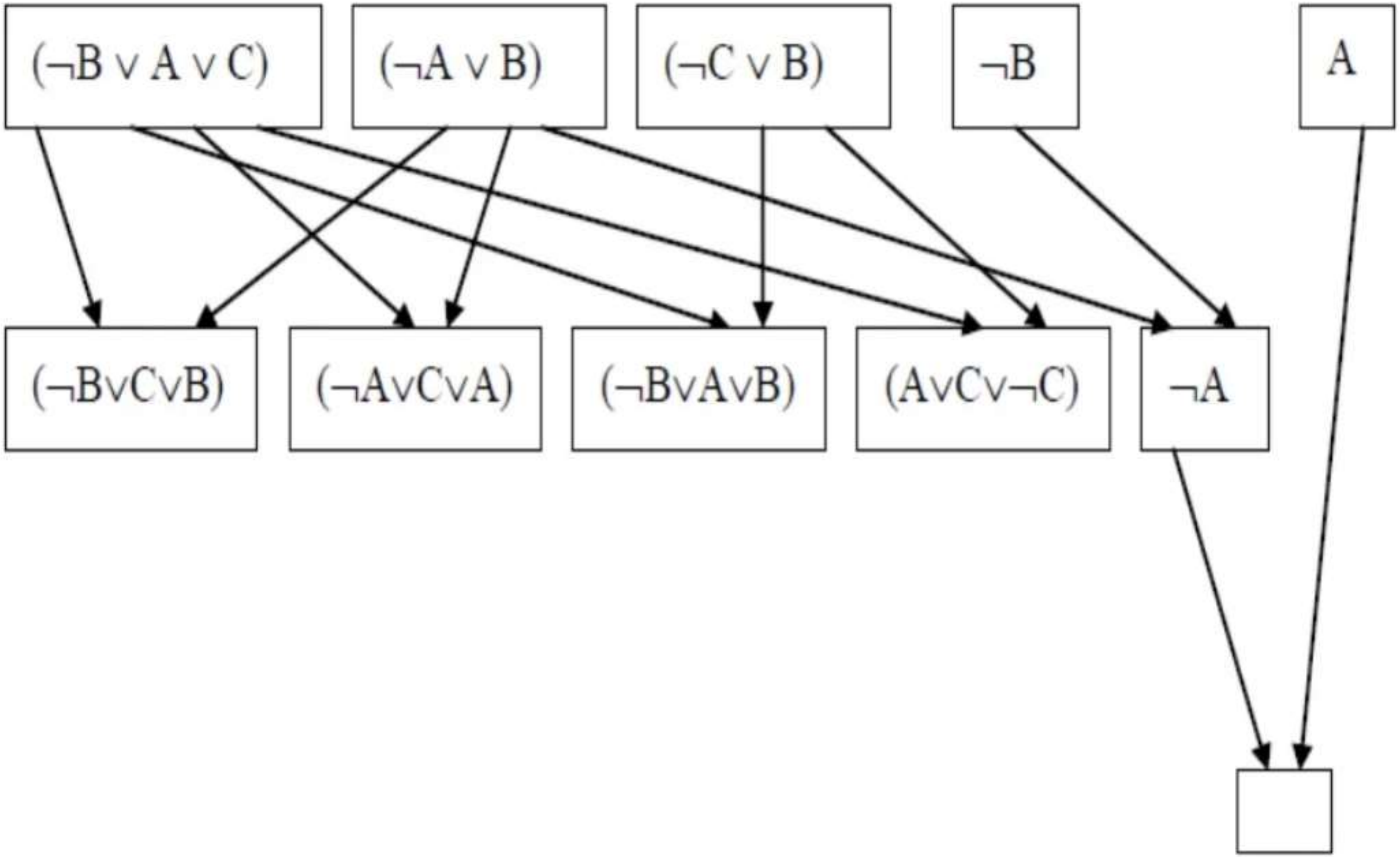
$(\neg B \vee A \vee C) \wedge ((\neg A \wedge \neg C) \vee B) \wedge \neg B$

$(\neg B \vee A \vee C) \wedge (\neg A \vee B) \wedge (\neg C \vee B) \wedge \neg B$

Add negation of sentence to be inferred from KB into KB

Now KB contains following sentences all in CNF
$(\neg B \vee A \vee C)$
$(\neg A \vee B)$
$(\neg C \vee B)$
$\neg B$
A (negation of conclusion to be proved)

Now use Resolution algorithm


(¬B ∨ A ∨ C)
(¬A ∨ B)
(¬C ∨ B)
¬B
A
(¬B∨C∨B)
(¬A∨C∨A)
(¬B∨A∨B)
(A∨C∨¬C)
¬A

Pros and cons of propositional logic

- Propositional logic is declarative
- Propositional logic is compositional:
  - meaning of $B \wedge P$ is derived from meaning of B and of P
- Meaning in propositional logic is context-independent
  - (unlike natural language, where meaning depends on context)
- Propositional logic has very limited expressive power (unlike natural language)

Comparison Between propositional logic and FOPL

- Propositional logic assumes the world contains facts, whereas first-order logic (like natural language) assumes the world contains:
  - Objects: people, houses, numbers, colors, baseball games, wars, …
  - Relations: red, round, prime, brother of, bigger than, part of, comes between,…
  - Functions: father of, best friend, one more than, plus, …
- The primary difference between PL and FOPL is their ontological commitment(**What** exists in the world — TRUTH)
  - PL: facts hold or do not hold.
  - FL : objects with relations between them that hold or do not hold

Predicate Logic

- propositional logic is best to illustrate the basic concepts of logic and knowledge-based agents.
- But, Propositional logic is limited in several ways.
  - Hard to represent information concisely.
  - Must deal with facts that are either TRUE or FALSE.
- **Predicate Logic!** a more powerful logic (use foundation of propositional logic) by adding more expressive concepts.

Predicates

A Predicate is a declarative sentence whose true/false value depends on one or more variables.

The statement “x is greater than 3” has two parts:

- the subject: x is the subject of the statement
- the predicate: “is greater than 3” (a property that the subject can have) .

We denote the statement “x is greater than 3” by P (x), where P is the predicate “is greater than 3” and x is the variable.

The statement P(x) is also called the value of propositional function P at x.

Assign a value to x, so P(x) becomes a proposition and has a truth value:

- P (5) is the statement “5 is greater than 3”, so P (5) is true.
- P (2) is the statement “2 is greater than 3”, so P (2) is false.

Prime(x) = “x is a prime number.”

- Prime(2) is true, since the only numbers that divide 2 are 1 and itself.
- Prime(9) is false, since 3 divides 9.

C(x, y)=“x is the capital of y”.

- C(Ottawa,Canada) is true.
- C(Buenos Aires,Brazil) is false.

E(x, y, z) = “x + y = z”.

- E(2, 3, 5) is ...
- E(4, 4, 17) is ...

- First-order logic (FOL) models the world in terms of
  - **Objects,** which are things with individual identities
  - **Properties** of objects that distinguish them from others
  - **Relations** that hold among sets of objects
  - **Functions,** which are a subset of relations where there is only one "value" for any given "input"
- Examples:
  - Objects: Students, lectures, companies, cars ...
  - Properties: blue, oval, even, large, ...
  - Relations: Brother, bigger-than, outside, part-of, has-color, occurs-after, owns, visits, precedes, ...
  - Functions: father-of, best-friend, second-half, one more-than ...

Representing knowledge in FOPL

- The basic syntactic elements of first order logics are the symbols.
- Formula in FOPL contains two types of Symbols. They are:

1. **User defined symbols**
   - Constants:
     - 3, John
     - Individuals
   - Functions:
     - f,g,h
     - mappings
   - Predicates:
     - P(x,y)
     - functions whose range is {True,False}

2. Logic defined symbols

- Variables:

  x,y,z

  Can be instantiated

- Logical Operators()
- Truth Symbols (TRUE, FALSE)

Quantifiers:

- ∀ for all
- ∃ there exists

Quantifiers in first order logic

- FOPL also provides variable quantifiers that allow the expression of properties for entire collections of objects instead of enumerating objects by name.
- Two types of quantifiers:
  - Universal quantifier ($\forall$): means "for all".
  - Existential quantification ($\exists$): means "there exists".
- Quantifiers are used with sentences containing variable symbols.
  - Let X be a variable symbol and P be a sentence.

    Then ($\forall$ X, P(X)) is a sentence and ($\exists$ X, P(X)) is a sentence.

- **Universal quantification:**
  - Often associated with English words "all", "everyone", "always", etc.
  - Syntax: ∀<Variables> <sentence>
  - E.g., Everyone at CAB is smart:

    $\forall x\ At(x, CAB) \Rightarrow Smart(x)$

    (we can also read this as "if X is at CAB, then X is smart)

- **$\forall x\ P(x)$ is true in a model M iff P(x) is true for all x in the model**
  - Roughly speaking, equivalent to the conjunction of instantiations of P
  - E.g.,: $At(Ram,CAB) \Rightarrow Smart(Ram) \wedge At(Hari,CAB) \Rightarrow Smart(Hari) \wedge \ldots$

- Typically, $\Rightarrow$ is the main connective with $\forall$
  - A universal quantifier is also equivalent to a set of implications over all objects
- Common mistake: using $\wedge$ as the main connective with $\forall$:
  - $\forall x\ At(x, CAB) \wedge Smart(x)$
  - Means "Everyone is at CAB and everyone is smart"

  - You rarely use universal quantification to make blanket statements about every individual in the world (because such statement is hardly true)

e.g., $\forall x\ human(x) \Rightarrow mortal(x)$
says, all humans are mortal
but, $\forall x\ human(x) \wedge mortal(x)$
say, everything is human and mortal

- **Existential quantification:**
  - Often associated with English words "someone", "sometimes", etc.
  - Syntax: ∃<variables> <sentence>
  - Example: Someone at CAB is smart:

    $\exists x \; At(x, CAB) \land Smart(x)$

- **$\exists x \; P(x)$ is true in a model m iff $P(x)$ is true for at least one x in the model**
  - Roughly speaking, equivalent to the disjunction of instantiations of P
  - $At(Ram, CAB) \land Smart(RAM) \lor At(Hari, CAB) \land Smart(Hari) \lor \ldots$

- Typically, $\wedge$ is the main connective with $\exists$
- Common mistake: using $\Rightarrow$ as the main connective with $\exists$
  - E.g.: $\exists x$ At(x, CAB) $\Rightarrow$ Smart(x) is true even if there is anyone who is not at CAB!

e.g., $\exists x\ bird(x) \wedge \neg flies(x)$

says, there is a bird that does not fly

but, $\exists x\ bird(x) \Rightarrow \neg flies(x)$

is also true for anything that is not a bird

Connections between All and Exists

- We can relate sentences involving $\forall$ and $\exists$ using **De Morgan's laws**:
  1. $(\forall x)\ \neg P(x) \leftrightarrow \neg(\exists x)\ P(x)$
  2. $\neg(\forall x)\ P(x) \leftrightarrow (\exists x)\ \neg P(x)$
  3. $(\forall x)\ P(x) \leftrightarrow \neg\ (\exists x)\ \neg P(x)$
  4. $(\exists x)\ P(x) \leftrightarrow \neg(\forall x)\ \neg P(x)$
- Examples
  1. All dogs don't like cats $\leftrightarrow$ No dogs like cats
  2. Not all dogs dance $\leftrightarrow$ There is a dog that doesn't dance
  3. All dogs sleep $\leftrightarrow$ There is no dog that doesn't sleep
  4. There is a dog that talks $\leftrightarrow$ Not all dogs can't talk

- Convert the following to the language of predicate logic.
  - Every apple is either green or yellow
  - No apple is blue
  - If an apple is green then its tasty
  - Every man likes a tasty apple
  - Some people like garlic
  - Fido is a dog and a good dog.
  - All basketball players are tall

- Every apple is either green or yellow

$$\forall X(apple(X) \Rightarrow green(X) \vee red(X))$$

- No apple is blue

$$\forall X(apple(x) \Rightarrow \neg blue(X))$$

- If an apple is green then its tasty

$$\forall X((apple(X) \wedge green(X)) \Rightarrow tasty(X))$$

- Every man likes a tasty apple

$$\forall X \exists Y(man(X) \wedge tastyApple(Y) \Rightarrow likes(X, Y)$$

- Some people like garlic

$$\exists X(person(X) \Rightarrow likes(X, garlic))$$

- Fido is a dog and a good dog.

$$dog(fido) \wedge gooddog(fido)$$

- All basketball players are tall

$$\forall X(basketballPlayer(X) \Rightarrow tall(X))$$

- Universal instantiation (UI):
  - UI Rule: Substitute ground term (term without variables) for the variables.
  - a universally quantified sentence can be replaced by the set of all possible instantiations.
  - After UI we discard the universally quantified sentence.

- For example: suppose our knowledge base contains just the sentences
    - $\forall$ x King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)   (all greedy kings are evil)
    - King(John)
    - Greedy(John)
    - Brother (Richard, John) .
  - Then we apply UI to the first sentence using all possible ground-term substitutions from the vocabulary of the knowledge base (in this case, {x/John} and {x/Richard }).
  - We obtain
    - King(John) $\wedge$ Greedy(John) $\Rightarrow$ Evil(John)
    - King(Richard) $\wedge$ Greedy(Richard) $\Rightarrow$ Evil(Richard) ,
  - and we discard the universally quantified sentence.

- Existential instantiation(EI):
  - EI rule: the variable is replaced by a single new constant symbol does not appear elsewhere in the knowledge base.
  - Basically, the existential sentence says there is some object satisfying a condition, and applying the existential instantiation rule just gives a name to that object.
  - So, Existential Instantiation can be applied once, and then the existentially quantified sentence can be discarded.
  - For example, we no longer need $\exists$ x Kill(x, Ram) once we have added the sentence Kill (Hari , Ram).

Simple Proof Example1

- Suppose we have the following sentences in the KB:
  - Anything that barks is a dog.
  - Fido barks.
  - prove that Fido is a dog.

- Step 1: State the facts you know in FOL
  - We know anything that barks is a dog. State this fact in FOL:

    a. $\forall x Barks(x) \Rightarrow Dog(x)$

    - This says "If x barks then x is a dog".
  - We know that Fido barks. State this fact also in FOL:

    b. Barks(Fido)

- Step 2: Remove all quantifiers
  - Apply the Universal Instantiation inference rule to remove the universal quantifier ($\forall$) in sentence a
  - The result is:

c. $Barks(Fido) \Rightarrow Dog(Fido)$

- Now KB in propositional form is:

  Barks(Fido)
  Barks(Fido) $\Rightarrow$ Dog(Fido)

- Step 3: See what inference rules can be applied.
  - Think about what we want to do: Eliminate the implication, leaving the sentence "Dog(Fido)"
  - How can we so this?
  - Use Modus Ponens

  D. Dog(Fido)

- **CNF Conversion summarized algorithm:**
  - Above descriptive steps of CNF conversion can be summarized in the following steps

1. Eliminate implications and bi-implications as in propositional case
2. Move negations inward using De Morgan's laws

   plus rewriting $\neg\forall xP$ as $\exists x\neg P$ and $\neg\exists xP$ as $\forall x\neg P$
3. Eliminate double negations
4. Rename bound variables if necessary so each only occurs once

   e.g. $\forall xP(x)\lor\exists xQ(x)$ becomes $\forall xP(x)\lor\exists yQ(y)$
5. Use equivalences to move quantifiers to the left

   e.g. $\forall xP(x)\land Q$ becomes $\forall x\,(P(x)\land Q)$ where x is not in Q

   e.g. $\forall xP(x)\land\exists yQ(y)$ becomes $\forall x\exists y(P(x)\land Q(y))$
6. Skolemise (replace each existentially quantified variable by a **new** term)

   $\exists xP(x)$ becomes $P(a_0)$ using a Skolem constant a_0 since $\exists x$ occurs at the outermost level

   $\forall x\exists yP(x,y)$ becomes $P(x,f_0(x))$ using a Skolem function f_0 since $\exists y$ occurs within $\forall x$
7. The formula now has only universal quantifiers and all are at the left of the formula: drop them
8. Use distribution laws to get CNF and then clausal form

The resolution inference rule:

- Two clauses, which are assumed to be standardized apart so that they share no variables, can be resolved if they contain complementary literals.
- Propositional literals are complementary if one is the negation of the other; first-order literals are complementary if one unifies with the negation of the other. Thus, we have

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{\text{SUBST}(\theta, \ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)}$$

Resolution Algorithm

- Algorithm:
  - Convert KB into first order logic expressions.
  - Convert knowledge base (FOPL logic expressions) into CNF
  - convert the negation of query into CNF and then add them into KB.
  - Repeatedly apply resolution to clauses or copies of clauses(a copy of a clause is the clause with all variables renamed) until either the empty clause is derived or no more clauses can be derived.
    - If the empty clause is derived , answer = Yes (query follows form knowledge base).
    - Otherwise answer = No (query does not follow from knowledge base)

Example of resolution refutation

- Example: Consider the following statements:
  - Everyone who loves all animal is loved by some one.
  - Jack is loves all animal
  - Query: Jack is loved by someone.

- KB in FOPL:
  - $\forall$ x [$\forall$ y Animal(y) $\Rightarrow$ Loves(x, y)] $\Rightarrow$ [$\exists$ y Loves(y, x)] .
  - $\forall$ y Animal(y) $\Rightarrow$ Loves(jack, y)
- Query in FOPL:
  - $\exists$y Animal(y) $\Rightarrow$ Loves(y, jack)
- Negatón of query:
  - $\forall$ y $\neg$ Animal(y) $\Rightarrow$ $\neg$ Loves(y, jack)

- KB and Query in CNF:
  - $[Animal(F(x)) \lor Loves(G(x), x)]$
  - $[\neg Loves(x, F(x)) \lor Loves(G(x), x)$
  - $\neg Animal(y) \lor Loves(jack, y)$
  - $Animal(y) \lor \neg Loves(y, jack)$

$\neg$ Animal(y) $\lor$ Loves(jack, y)   Animal(y) $\lor$ $\neg$ Loves(y, jack)

Hence jack is loved by someone

Example of resolution refutation

KB:

1. Anyone passing his history exams and winning the lottery is happy.

2. Anyone who studies or is lucky can pass all his exams.

3. John did not study but John is lucky.

4. Anyone who is lucky wins the lottery.

query: “Is John happy?”.
Use the resolution refutation algorithm to answer the given query.

(a) Translate the following four English sentences to first order logic (FOL).

1. Anyone passing his history exams and winning the lottery is happy.

2. Anyone who studies or is lucky can pass all his exams.

3. John did not study but John is lucky.

4. Anyone who is lucky wins the lottery.

(b) Convert them to conjunctive normal form (CNF).

(c) Answer the query "Is John happy?". Use the resolution refutation algorithm.

(a) Translate the following four English sentences to first order logic (FOL).

1. Anyone passing his history exams and winning the lottery is happy.

$$\forall x \; Pass(x, HistoryExam) \wedge Win(x, Lottery) \Rightarrow Happy(x)$$

2. Anyone who studies or is lucky can pass all his exams.

3. John did not study but John is lucky.

4. Anyone who is lucky wins the lottery.

1. Anyone passing his history exams and winning the lottery is happy.

$$\forall x\ Pass(x, HistoryExam) \wedge Win(x, Lottery) \Rightarrow Happy(x)$$

2. Anyone who studies or is lucky can pass all his exams.

$$\forall x \forall y\ Study(x) \vee Lucky(x) \Rightarrow Pass(x, y)$$

3. John did not study but John is lucky.

4. Anyone who is lucky wins the lottery.

1. Anyone passing his history exams and winning the lottery is happy.

$$\forall x\ Pass(x, HistoryExam) \wedge Win(x, Lottery) \Rightarrow Happy(x)$$

2. Anyone who studies or is lucky can pass all his exams.

$$\forall x \forall y\ Study(x) \vee Lucky(x) \Rightarrow Pass(x, y)$$

3. John did not study but John is lucky.

$$\neg Study(John) \wedge Lucky(John)$$

4. Anyone who is lucky wins the lottery.

1. Anyone passing his history exams and winning the lottery is happy.

$$\forall x\, Pass(x, HistoryExam) \wedge Win(x, Lottery) \Rightarrow Happy(x)$$

2. Anyone who studies or is lucky can pass all his exams.

$$\forall x \forall y\, Study(x) \vee Lucky(x) \Rightarrow Pass(x, y)$$

3. John did not study but John is lucky.

$$\neg Study(John) \wedge Lucky(John)$$

4. Anyone who is lucky wins the lottery.

$$\forall x\, Lucky(x) \Rightarrow Win(x, Lottery)$$

(b) Convert them to conjunctive normal form (CNF).

1 $\forall x\ Pass(x, HistoryExam) \wedge Win(x, Lottery) \Rightarrow Happy(x)$

2 $\forall x \forall y\ Study(x) \vee Lucky(x) \Rightarrow Pass(x, y)$

3 $\neg Study(John) \wedge Lucky(John)$

4 $\forall x\ Lucky(x) \Rightarrow Win(x, Lottery)$

First: Implication elimination

1 $\neg[Pass(x, HistoryExam) \wedge Win(x, Lottery)] \vee Happy(x)$

2 $\neg[Study(x) \vee Lucky(x)] \vee Pass(x, y)$

3 $\neg Study(John) \wedge Lucky(John)$

4 $\neg Lucky(x) \vee Win(x, Lottery)$

Then: drop the "for all" quantifiers

1 $$[\neg Pass(x, HistoryExam) \vee \neg Win(x, Lottery)] \vee Happy(x)$$

2 $$[\neg Study(x) \wedge \neg Lucky(x)] \vee Pass(x, y)$$

3 $$\neg Study(John) \wedge Lucky(John)$$

4 $$\neg Lucky(x) \vee Win(x, Lottery)$$

Then: move the negation inwards

1 $\neg Pass(x, HistoryExam) \vee \neg Win(x, Lottery) \vee Happy(x)$

2 $[\neg Study(x) \vee Pass(x, y)] \wedge [\neg Lucky(x) \vee Pass(x, y)]$

3 $\neg Study(John) \wedge Lucky(John)$

4 $\neg Lucky(x) \vee Win(x, Lottery)$

Then: distribute the ”ors”

1 $\neg Pass(x, HistoryExam) \vee \neg Win(x, Lottery) \vee Happy(x)$

2a $\neg Study(x) \vee Pass(x, y)$

2b $\neg Lucky(x) \vee Pass(x, y)$

3a $\neg Study(John)$

3b $Lucky(John)$

4 $\neg Lucky(x) \vee Win(x, Lottery)$

Then: separate the ”and” sentences into individual sentences

(c) Query and resolution refutation

1	$\neg Pass(x, HistoryExam) \vee \neg Win(x, Lottery) \vee Happy(x)$
2a	$\neg Study(x) \vee Pass(x, y)$
2b	$\neg Lucky(x) \vee Pass(x, y)$
3a	$\neg Study(John)$
3b	$Lucky(John)$
4	$\neg Lucky(x) \vee Win(x, Lottery)$

Knowledge Base (KB)

Note: Construct the resolution tree for the clauses given below as in propositional logic but use the predicate resolution rule instead of resolution rule of propositional logic.

1 $\neg Pass(x, HistoryExam) \vee \neg Win(x, Lottery) \vee Happy(x)$

2a $\neg Study(x) \vee Pass(x, y)$

2b $\neg Lucky(x) \vee Pass(x, y)$

3a $\neg Study(John)$

3b $Lucky(John)$

4 $\neg Lucky(x) \vee Win(x, Lottery)$

$\boxed{Q \quad \neg Happy(John)}$

The negation of our query

$$\boxed{1 \quad \neg Pass(x, HistoryExam) \vee \neg Win(x, Lottery) \vee Happy(x)}$$

2a $\neg Study(x) \vee Pass(x, y)$

2b $\neg Lucky(x) \vee Pass(x, y)$

{x/John}

3a $\neg Study(John)$

3b $Lucky(John)$

4 $\neg Lucky(x) \vee Win(x, Lottery)$

$$\boxed{Q \quad \neg Happy(John)}$$

1 $\neg Pass(x, HistoryExam) \vee \neg Win(x, Lottery) \vee Happy(x)$

2a $\neg Study(x) \vee Pass(x, y)$

2b $\neg Lucky(x) \vee Pass(x, y)$

3a $\neg Study(John)$

{x/John; y/HistoryExam}

3b $Lucky(John)$

4 $\neg Lucky(x) \vee Win(x, Lottery)$

$\neg Pass(John, HistoryExam) \vee \neg Win(John, Lottery)$ 5

Q $\neg Happy(John)$

1 $\neg Pass(x, HistoryExam) \vee \neg Win(x, Lottery) \vee Happy(x)$

2a $\neg Study(x) \vee Pass(x, y)$

2b $\neg Lucky(x) \vee Pass(x, y)$

3a $\neg Study(John)$

3b $Lucky(John)$

$\boxed{4 \quad \neg Lucky(x) \vee Win(x, Lottery)}$ {x/John}

$\neg Pass(John, HistoryExam) \vee \neg Win(John, Lottery)$ 5

$\boxed{\neg Lucky(John) \vee \neg Win(John, Lottery) \quad 6}$

1 $\neg Pass(x, HistoryExam) \vee \neg Win(x, Lottery) \vee Happy(x)$

2a $\neg Study(x) \vee Pass(x, y)$

$\neg Lucky(John)$ 7

3a $\neg Study(John)$

3b $Lucky(John)$

4 $\neg Lucky(x) \vee Win(x, Lottery)$ {x/John}

$\neg Pass(John, HistoryExam) \vee \neg Win(John, Lottery)$ 5

$\neg Lucky(John) \vee \neg Win(John, Lottery)$ 6

1 $\neg Pass(x, HistoryExam) \vee \neg Win(x, Lottery) \vee Happy(x)$

2a $\neg Study(x) \vee Pass(x, y)$

$\neg Lucky(John)$ 7

3a $\neg Study(John)$

3b $Lucky(John)$

$\varnothing$

4 $\neg Lucky(x) \vee Win(x, Lottery)$

$\neg Pass(John, HistoryExam) \vee \neg Win(John, Lottery)$ 5

$\neg Lucky(John) \vee \neg Win(John, Lottery)$ 6

Forward Chaining

Scenario: Determine the color of a pet named Fritz based on given facts and rules.
Facts:

- Fritz croaks.
- Fritz eats flies.

Rules:

1. If X croaks and X eats flies, then X is a frog.
2. If X is a frog, then X is green.
3. If X chirps and X sings, then X is a canary.
4. If X is a canary, then X is blue.

Process:

1. Start with the known facts: Fritz croaks and Fritz eats flies.
2. Apply Rule 1: Since Fritz meets the conditions (croaks and eats flies), we conclude that Fritz is a frog.
3. Apply Rule 2: Now that we know Fritz is a frog, we can conclude that Fritz is green.

In this example, forward chaining starts with the available data (facts) and applies rules to derive new information until reaching a conclusion.

Backward Chaining

Scenario: Prove that John is the tallest boy in his class.

Goal: Show that John is the tallest boy.

Facts and Rules:

- John is taller than Kim.
- Kim is shorter than everyone else in the class except John.
- John is a boy.
- Kim is a girl.

Process:

1. Start with the goal: Is John the tallest boy?
2. Assume John is the tallest and check if this leads to contradictions or supports from known facts:Check if John being taller than Kim supports the goal.
3. Since all conditions for the goal are satisfied, we conclude that John is indeed the tallest boy in his class.

In this example, backward chaining begins with the goal and works backward through the rules to verify if it can be supported by existing facts.

Unification

- Unification is the process of finding substitutions that make different logical expressions look identical.
- takes two sentences and returns a list of substitutions(unifier) to make two sentences match, or failure if no match possible.
- i.e., UNIFY(p, q)=θ , are matched where θ is the list of substitutions in p and q

- **Unification rules:**

1. Function symbols and predicate symbols must have identical names and number of arguments.
2. Constant symbols unify with only identical constant symbols.
3. Variables unify with other variable symbols, constant symbols or function symbols
4. Variable symbols may not be unified with other terms in which the variable itself occurs.
   - For example: x can not unify with G(x) since this will lead to G(G(G(….G(x))))

- Example: unification

p	q	θ
Knows(John,x)	Knows(John,Jane)	{x/Jane}
Knows(John,x)	Knows(y,OJ)	{x/OJ,y/John}
Knows(John,x)	Knows(y,Mother(y))	{y/John,x/Mother(John)}}
Knows(John,x)	Knows(x,OJ)	{fail}

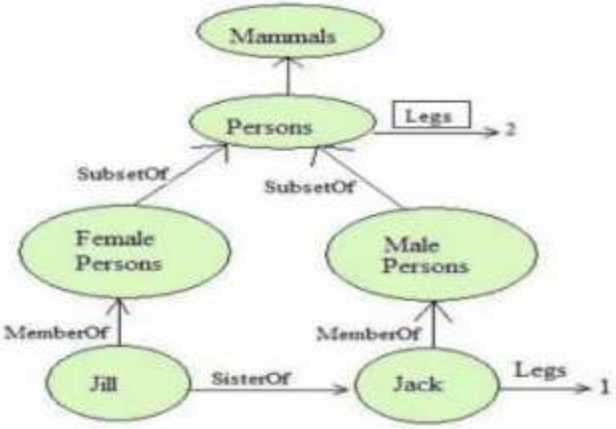
- Last unification is failed due to overlap of variables
- i.e., x can not take the values of John and OJ at the same time.
- We can avoid this problem(name clashes) by renaming (standardizing apart)
- E.g., Unify{Knows(John,x) Knows(z,OJ) } = {x/OJ, z/John}

Semantic Network

- A **semantic net** (or semantic network) is a knowledge representation technique used for propositional information. So it is also called a propositional net.
- Mathematically a semantic net can be defined as a labeled directed graph.
- consist of:
  - nodes,
  - links (edges) and
  - link labels.

- Nodes:
  - In the semantic network diagram, 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, and
- link labels:
  - specify particular relations. Relationships provide the basic structure for organizing knowledge.
  - The objects and relations involved need not be so concrete.
- As nodes are associated with other nodes semantic nets are also referred to as associative nets.

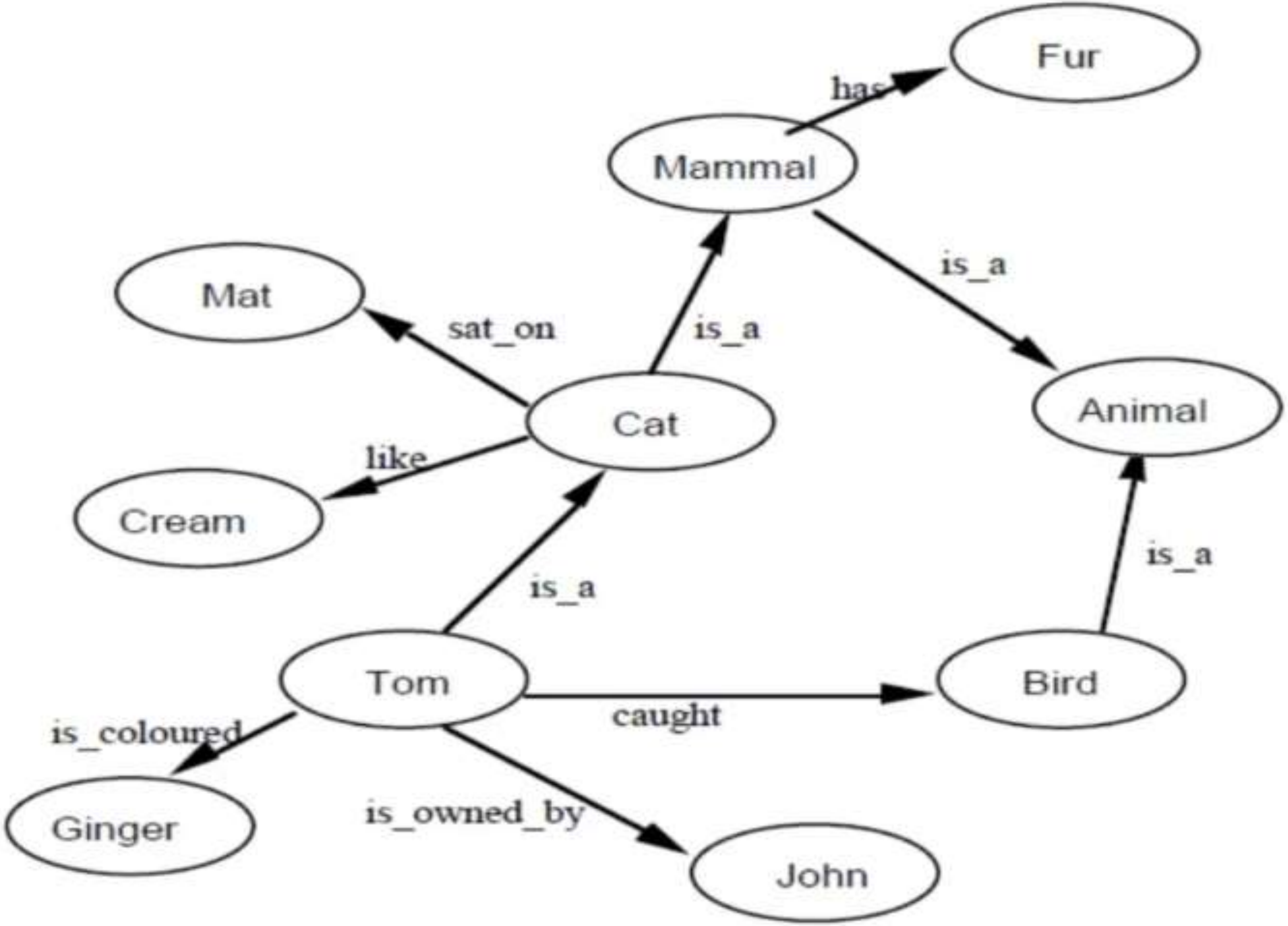
Semantic Network



- In the above figure all the objects are within ovals and connected using labelled arcs.
- Note that there is a link between Jill and FemalePersons with label MemberOf. Simlarly there is a MemberOf link between Jack and MalePersons and SisterOf link between Jill and Jack.
- The MemberOf link between Jill and FemalePersons indicates that Jill belongs to the category of female persons.

Example1: semantic network

- Represent the following fact in semantic network
  - Tom is a cat.
  - Tom caught a bird.
  - Tom is owned by John.
  - Tom is ginger in color.
  - Cats like cream.
  - The cat sat on the mat.
  - A cat is a mammal.
  - A bird is an animal.
  - All mammals are animals.
  - Mammals have fur.



Fur
has
Mammal
is_a
Mat
sat_on
is_a
Cat
Animal
like
Cream
is_a
is_a
Tom
Bird
caught
is_coloured
Ginger
is_owned_by
John

Bayes' Rule

Bayes' theorem is stated mathematically as the following equation:

$$P(A \mid B) = P(B \mid A) * P(A) / P(B)$$

where A and B are events and $P(B) \neq 0$

$P(A \mid B)$ is a conditional probability: the likelihood of event A occurring given that B is true.
$P(B \mid A)$ is also a conditional probability: the likelihood of event B occurring given that As true.
$P(A)$ and $P(B)$ are the probabilities of observing A and B independently of each other; this is known as the marginal probability.

Bayes Rule **More Simply**

It tells us how often A happens given that B happens, written P(A|B), when we know how often B happens given that A happens, written P(B|A) , and how likely A and B are on their own.

P(A|B) is "Probability of A given B", the probability of A given that B happens

P(A) is Probability of A

P(B|A) is "Probability of B given A", the probability of B given that A happens

P(B) is Probability of B

Naive Bayes Rule

The term "naive" in Naïve Bayes arises from its strong and often unrealistic assumption that all features are independent given the class label.

While this may not hold true in many situations, it allows for efficient computation and has proven effective in various applications, particularly when dealing with large datasets where computational efficiency is crucial.

German Swiss Speaker Example

There are about 8.4 million people living in Switzerland. About 64 % of them speak German. There are about 7500 million people on earth.

If some aliens randomly beam up an earthling, what are the chances that he is a German speaking Swiss?

We have the events

S: being Swiss

GS: German Speaking

$P(S)=8.4/7500=0.00112$

If we know that somebody is Swiss, the probability of speaking German is 0.64. This corresponds to the conditional probability

$P(GS|S)=0.64$

So the probability of the earthling being Swiss and speaking German, can be calculated by the formula:

$P(GS|S)=P(GS\cap S)P(S)$

inserting the values from above gives us:

$0.64=P(GS\cap S)/0.00112$

and

$P(GS\cap S)=0.0007168$

So our aliens end up with a chance of 0.07168 % of getting a German speaking Swiss person.

Bayesian Belief Network

Bayesian belief network is key computer technology for dealing with probabilistic events and to solve a problem which has uncertainty. We can define a Bayesian network as:

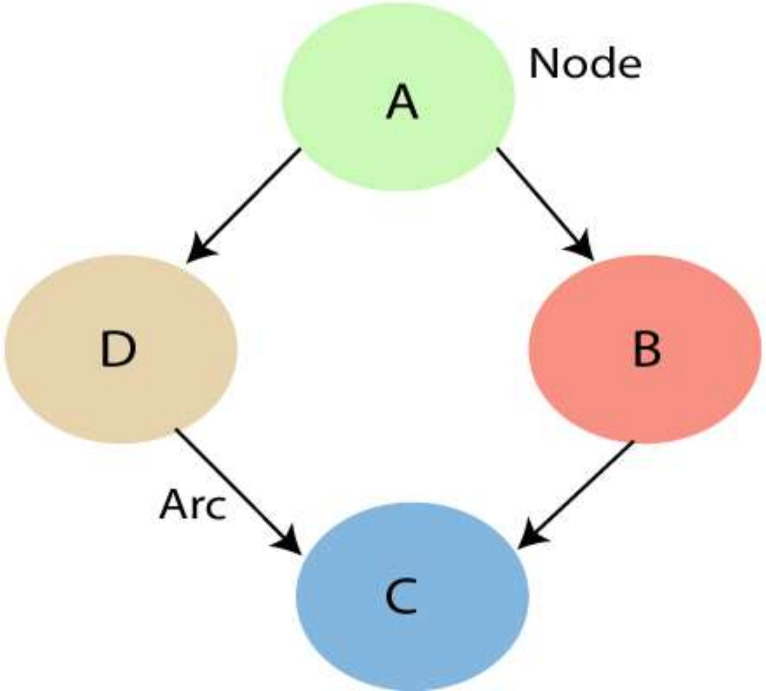
"A Bayesian network is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph."

It is also called a **Bayes network, belief network, decision network**, or **Bayesian model**.

Bayesian networks are probabilistic, because these networks are built from a **probability distribution**, and also use probability theory for prediction and anomaly detection.

The generalized form of Bayesian network that represents and solve decision problems under uncertain knowledge is known as an **Influence diagram**.

A Bayesian network graph is made up of nodes and Arcs (directed links), where:



- Each **node** corresponds to the random variables, and a variable can be **continuous** or **discrete**.
- **Arc or directed arrows** represent the causal relationship or conditional probabilities between random variables. These directed links or arrows connect the pair of nodes in the graph.
  These links represent that one node directly influence the other node, and if there is no directed link that means that nodes are independent with each other
  - **In the above diagram, A, B, C, and D are random variables represented by the nodes of the network graph.**
  - **If we are considering node B, which is connected with node A by a directed arrow, then node A is called the parent of Node B.**
  - **Node C is independent of node A.**

Each node in the Bayesian network has condition probability distribution

$P(X_i | Parent(X_i))$, which determines the effect of the parent on that node.

Bayesian network is based on Joint probability distribution and conditional probability. So let's first understand the joint probability distribution:

If we have variables x1, x2, x3,....., xn, then the probabilities of a different combination of x1, x2, x3.. xn, are known as Joint probability distribution.

$P[x_1, x_2, x_3,....., x_n]$, it can be written as the following way in terms of the joint probability distribution.

$= P[x_1| x_2, x_3,....., x_n]P[x_2, x_3,....., x_n]$

$= P[x_1| x_2, x_3,....., x_n]P[x_2|x_3,....., x_n]....P[x_{n-1}|x_n]P[x_n]$.

In general for each variable Xi, we can write the equation as:

$P(X_i|X_{i-1},........., X_1) = P(X_i |Parents(X_i))$

Example

Harry installed a new burglar alarm at his home to detect burglary. The alarm reliably responds at detecting a burglary but also responds for minor earthquakes. Harry has two neighbors David and Sophia, who have taken a responsibility to inform Harry at work when they hear the alarm. David always calls Harry when he hears the alarm, but sometimes he got confused with the phone ringing and calls at that time too. On the other hand, Sophia likes to listen to high music, so sometimes she misses to hear the alarm. Here we would like to compute the probability of Burglary Alarm.

Problem:

Calculate the probability that alarm has sounded, but there is neither a burglary, nor an earthquake occurred, and David and Sophia both called the Harry.

Solution:

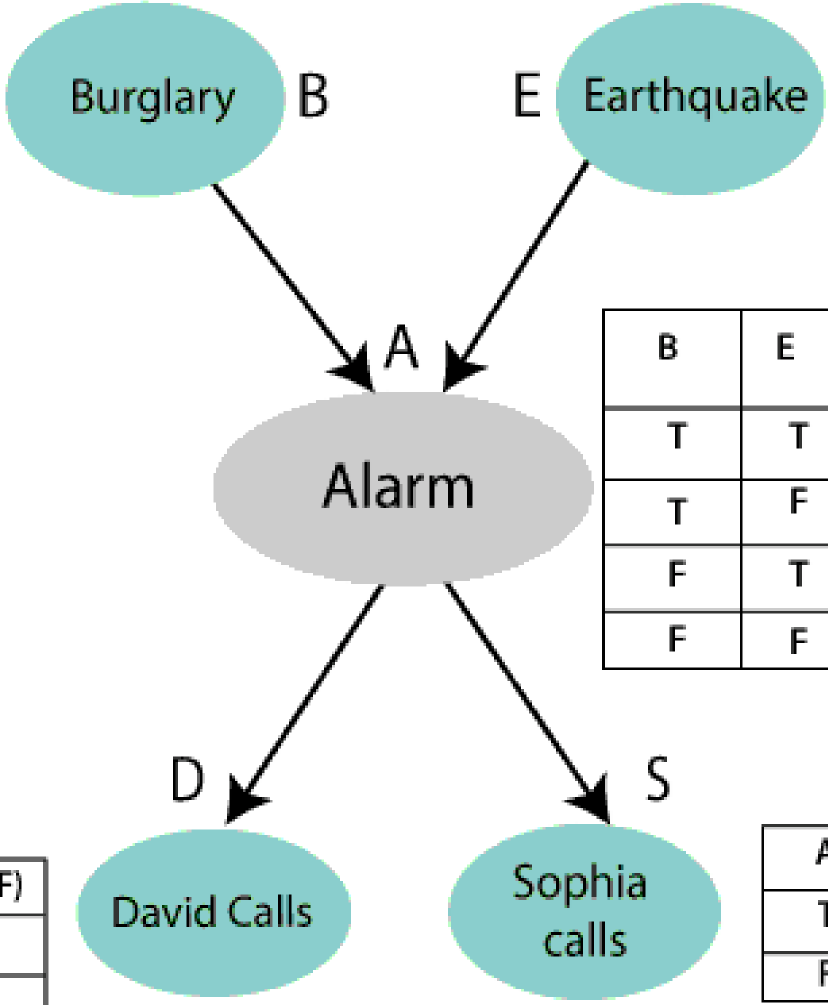
- The Bayesian network for the above problem is given below. The network structure is showing that burglary and earthquake is the parent node of the alarm and directly affecting the probability of alarm's going off, but David and Sophia's calls depend on alarm probability.
- The network is representing that our assumptions do not directly perceive the burglary and also do not notice the minor earthquake, and they also not confer before calling.
- The conditional distributions for each node are given as conditional probabilities table or CPT.
- Each row in the CPT must be sum to 1 because all the entries in the table represent an exhaustive set of cases for the variable.
- In CPT, a boolean variable with k boolean parents contains 2^K probabilities. Hence, if there are two parents, then CPT will contain 4 probability values

List of all events occurring in this network:

- **Burglary (B)**
- **Earthquake(E)**
- **Alarm(A)**
- **David Calls(D)**
- **Sophia calls(S)**

T	0.002
F	0.998

T	0.001
F	0.999



B	E	P(A=T)	P(A=F)
T	T	0.94	0.06
T	F	0.95	0.04
F	T	0.69	0.69
F	F	0.999	0.999

A	P (D=T)	P (D=F)
T	0.91	0.09
F	0.05	0.95

A	P (S=T)	P (S=F)
T	0.75	0.25
F	0.02	0.98

P(B= True) = 0.002, which is the probability of burglary.

P(B= False)= 0.998, which is the probability of no burglary.

P(E= True)= 0.001, which is the probability of a minor earthquake

P(E= False)= 0.999, Which is the probability that an earthquake not occurred.

Conditional probability table for Alarm A:

The Conditional probability of Alarm A depends on Burglar and earthquake:

B	E	P(A= True)	P(A= False)
True	True	0.94	0.06
True	False	0.95	0.04
False	True	0.31	0.69
False	False	0.001	0.999

Conditional probability table for David Calls:

The Conditional probability of David that he will call depends on the probability of Alarm.

A	P(D= True)	P(D= False)
True	0.91	0.09
False	0.05	0.95

Conditional probability table for Sophia Calls:

The Conditional probability of Sophia that she calls is depending on its Parent Node "Alarm."

A	P(S= True)	P(S= False)
True	0.75	0.25
False	0.02	0.98

From the formula of joint distribution, we can write the problem statement in the form of probability distribution:

P(S, D, A, ¬B, ¬E) = P (S|A) *P (D|A)*P (A|¬B ^ ¬E) *P (¬B) *P (¬E).

= 0.75* 0.91* 0.001* 0.998*0.999

= 0.00068045.

Hence, a Bayesian network can answer any query about the domain by using Joint distribution.

MCQ

1. Knowledge and reasoning also play a crucial role in dealing with ____________________ environment.

a) Completely Observable

b) Partially Observable

c) Neither Completely nor Partially Observable

d) Only Completely and Partially Observable

Answer: b

2. A) Knowledge base (KB) is consists of set of statements.

B) Inference is deriving a new sentence from the KB.

Choose the correct option.

a) A is true, B is true

b) A is false, B is false

c) A is true, B is false

d) A is false, B is true

Answer: a

3. Wumpus World is a classic problem, best example of _______

a) Single player Game

b) Two player Game

c) Reasoning with Knowledge

d) Knowledge based Game

Answwer: c

4. ‘α |= β ‘(to mean that the sentence α entails the sentence β) if and only if, in every model in which α is _____ β is also _____

a) True, true

b) True, false

c) False, true

d) False, false

Answer: a

5. Which is not Familiar Connectives in First Order Logic?

a) and

b) iff

c) or

d) not

Answer: d

6. Inference algorithm is complete only if ______________

a) It can derive any sentence

b) It can derive any sentence that is an entailed version

c) It is truth preserving

d) It can derive any sentence that is an entailed version & It is truth preserving

Answer: d

7. What among the following could the universal instantiation of ___________

For all x King(x) ^ Greedy(x) => Evil(x)

a) King(John) ^ Greedy(John) => Evil(John)

b) King(y) ^ Greedy(y) => Evil(y)

c) King(Richard) ^ Greedy(Richard) => Evil(Richard)

d) All of the mentioned

Answer: d

8. Lifted inference rules require finding substitutions that make different logical expressions looks identical.

a) Existential Instantiation

b) Universal Instantiation

c) Unification

d) Modus Ponen

Answer: c

9. Which of the following is not the style of inference?

a) Forward Chaining

b) Backward Chaining

c) Resolution Refutation

d) Modus Ponen

Answer: d

10. Which among the following could the Existential instantiation of ∃x Crown(x) ^ OnHead(x, Johnny)?

a) Crown(John) ^ OnHead(John, Jonny)

b) Crown(y) ^ OnHead(y, y, x)

c) Crown(x) ^ OnHead(x, Jonny)

d) None of the mentioned

Answer: a

11. Translate the following statement into FOL.

“For every a, if a is a PhD student, then a has a master degree”

a) ∀ a PhD(a) -> Master(a)

b) ∃ a PhD(a) -> Master(a)

c) A is true, B is true

d) A is false, B is false

Answer: a

12. What are Semantic Networks?

a) A way of representing knowledge

b) Data Structure

c) Data Type

d) None of the mentioned

Answer: a

13. Graph used to represent semantic network is _____________

a) Undirected graph

b) Directed graph

c) Directed Acyclic graph (DAG)

d) Directed complete graph

Answer: b

14. Which of the following elements constitutes the frame structure?

a) Facts or Data

b) Procedures and default values

c) Frame names

d) Frame reference in hierarchy

Answer: a

15. What is the primary purpose of propositional logic?

A) To represent and manipulate numerical data

B) To reason about propositions that can be either true or false

C) To perform calculations with real numbers

D) To create graphical representations of data

Answer: b

16. Which of the following statements is true about the resolution rule in propositional logic?

A) It can only be applied to disjunctive clauses.

B) It requires two clauses with a common literal.

C) It can derive new clauses by eliminating a variable.

D) It is used exclusively for proving theorems in predicate logic.

Answer: C

17. In resolution refutation, what must be done to prove that a statement P is false?

A) Add P to the set of premises and derive a contradiction.

B) Negate P and add it to the set of premises.

C) Use only positive clauses to derive P.

D) Show that P can be derived from existing premises.

Answer: B

18. What is the primary assumption made by the Naïve Bayes classifier regarding features?

A) Features are dependent on each other.

B) Features are conditionally independent given the class label.

C) Features have a normal distribution.

D) Features are uniformly distributed.

Answer: B

19. In a Bayesian network, what does each node represent?

A) A random variable

B) A deterministic function

C) A fixed value

D) An observation

Answer: A

20. Which of the following is a common application of Naïve Bayes classifiers?

A) Image recognition

B) Spam detection in emails

C) Time series forecasting

D) Reinforcement learning

Answer: B

21. What is the purpose of the conditional probability tables (CPTs) in a Bayesian network?

A) To store prior probabilities of nodes

B) To define the relationships between parent and child nodes

C) To calculate the joint probability distribution

D) To represent the overall structure of the network

Answer: B

22. When using Naïve Bayes for classification, what type of probability does it compute to make predictions?

A) Joint probability

B) Marginal probability

C) Posterior probability

D) Prior probability

Answer: C

23. In first-order predicate logic, what does a predicate represent?

A) A constant value

B) A relationship or property of objects

C) A logical connective

D) A quantifier

Answer: B

24. What is the main advantage of using semantic networks over traditional databases?

A) They require less memory.

B) They allow for easy representation of complex relationships.

C) They are faster for numerical computations.

D) They can only represent simple facts.

Answer: B

25. What is a belief network primarily used for?

A) Storing large datasets

B) Representing conditional dependencies among random variables

C) Performing arithmetic calculations

D) Visualizing data trends

Answer: B

26. What does inference in a belief network typically involve?

A) Summing all possible outcomes

B) Calculating joint probabilities for all variables

C) Updating beliefs based on new evidence

D) Finding deterministic relationships between variables

Answer: C

27. What is the primary function of logical connectives in propositional logic?

A) To represent numerical values

B) To construct complex sentences from simpler ones

C) To define the truth values of propositions

D) To eliminate redundancy in statements

Answer: B

28. Which of the following forms represents a conjunction of disjunctions of literals?

A) Conjunctive Normal Form (CNF)

B) Disjunctive Normal Form (DNF)

C) Normal Form

D) All of the mentioned

Answer: A

29. Which of the following is an example of a contradiction?

A) $p \vee {\sim}p$
B) $p \wedge {\sim}p$
C) $p \rightarrow p$
D) $p \leftrightarrow p$

30. The contrapositive of the statement "If p, then q" is:

A) If q then p
B) If ~p, then ~q
C) If ~q then ~p
D) If p then ~q