10/10/22

know

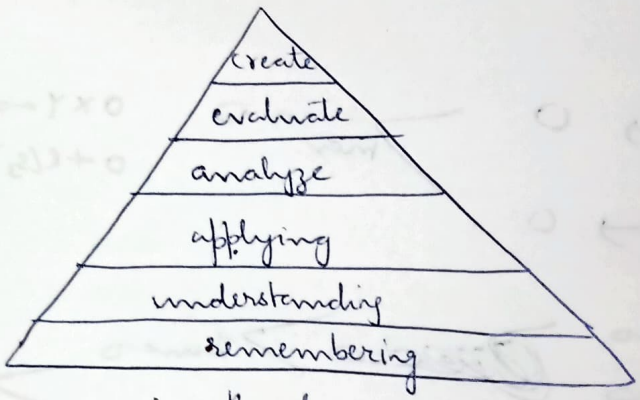
UNIT-2

Knowledge Representation

- Knowledge 1) It can be defined as the body of facts and principles accumulated by humankind or it can be understood as the fact or state of knowing.

2) Knowledge is having familiarity with language, concepts, procedures, rules, ideas, abstractions, places, customs, facts, associations coupled with an ability to use these notions effectively in modelling different aspects of the world.

3) The meaning of knowledge is closely related to the meaning of intelligence eg: BLOOM's TEXANOMY → Cognitive (Related to Mind)
→ Psychosensory



4) Intelligence requires the possession of an access to knowledge. A common way to represent knowledge external to a computer or human is in the form of written language.

5) For eg: Ramu is tall. This expresses a simple fact which is the attribute possessed by a person.

For eg: Ramu loves his mother. It is a complex binary relation b/w 2 persons. There are 2 entities related to representation of knowledge in AI.

① Fact: Fact is the truth in some relavant world. These are the things to be represented.

② Representation of facts: It must be done in some chosen formalism. There are the things that we will actually ~~are going~~ be able to manipulate.

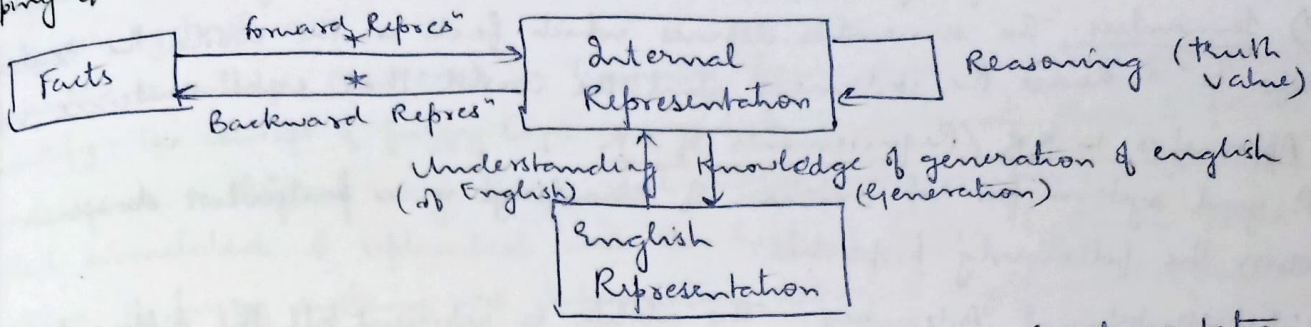
6) The structuring of these entities are done at 2 levels:

① Knowledge level: Facts including each agent's behaviours & current goals are described at this level.

② Symbol level: The Representation of objects at the knowledge level are defined in terms of symbols that can be manipulated by programs.

LISP, PROLOG → Tools

• Mapping b/w Facts & Representations



Consider the statement, Markens is a man. Using the logical representation, (x)?

① man(Marcus)
↳ man(x)

Suppose we have a logical representation which says all men have legs.

② ∀x : man(x) → haslegs(x)

③ haslegs(marcus) ← from ① & ②

This is called Deductive mathematical reasoning. (also Inductive)

Using backward mapping, we can say that Markens has legs. for eg: 2 statements are there: (i) All men have legs (ii) every man has legs.

(example of many-many relationship)

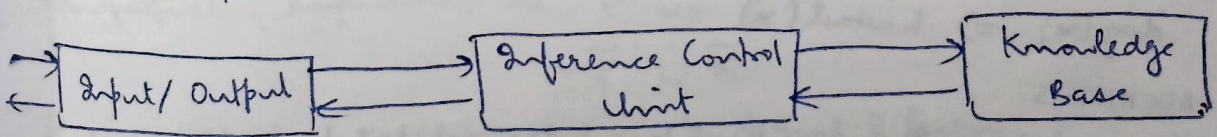
Conclusion: Every man has atleast 1 leg.

Note: We must 1st decide what facts the sentences represent & then convert those facts into the new representation.

12/10/22 Unit 2 → Richard Night.

• Knowledge Based System – These are all system that depends on rich base of knowledge to perform difficult tasks. It includes work in vision learning, general problem solving and natural language understandings. The systems get their power from the expert knowledge that has been coded into facts, heuristics & procedures. The knowledge is stored in a knowledge base that is separate from the control & inferencing components. It is possible to add new knowledge or refine existing knowledge without re-compiling the control & inferencing programs.

• Components of Knowledge Based System



• Representation of Knowledge

The objective of a knowledge Representation is to express knowledge in computer tractable form so that it can be used to enable our AI agents to perform as per the ~~user~~ expectations.

A knowledge Rep. language is defined by 2 aspects:

(1) Syntax - The syntax of a language defines which configurations of the

components of a language constitutes valid sentences.

(2) Semantics: The semantics defines which facts in the world the sentences refer to & hence the statement about the world that each sentence makes

- Approaches to KR / Requirements of KR

A good system for representation of knowledge in a particular domain should possess the following properties:

(1) Representational Adequacy - The ability to represent all the different kinds of knowledge that might be needed in that domain

(2) Inferencial Adequacy - The ability to manipulate the representational structures in such a way as to derive new structures corresponding to new knowledge inferred from old ones.

(3) Inferential efficiency - The ability to incorporate additional information into the knowledge structure which can be used to focus the attention on the inference mechanisms in the most promising directions.

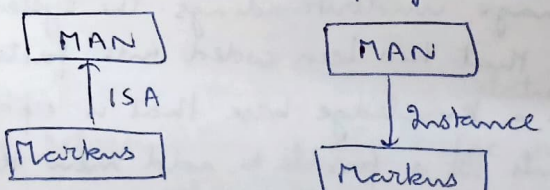
(4) Acquisitional efficiency - The ability to acquire new information easily. Ideally the agents should be able to control its own model acquisition. But direct insertion of information by knowledge engineer would be acceptable.

- Techniques of KR

(1) Simple Relational Knowledge - It is used in database systems to represent declarative facts. Declarative knowledge is passive knowledge expressed as statements or facts about the world.

(2) Inheritable knowledge - It uses the concept of Property Inheritance

2 imp. Characteristics =>



(3) Inferential Knowledge - It uses the concept of standard logical rules of Inference. 2 imp rules: Modesponence, Modestolence.

eg: $\forall x : dog(x) \Rightarrow hastail(x)$

$x = :SOCRATES$

(4) Procedural Knowledge - It is compiled knowledge related to performance of some tasks.
↳ eg: To solve Algebraic equations

- Fuzzy logic
↳ Unclear

(1) Fuzzy System - They include fuzzy logic & fuzzy set theory.

Knowledge exists in 2 distinct forms:

(1) Objective knowledge that exists in mathematical form & is used to represent the engineering problems.

(2) Subjective knowledge that exists in linguistic form & is usually impossible to quantify. The concept of fuzzy logic is used to co-ordinate these 2 forms of knowledge in a logical way. Many real world problems have been modelled simulated & replicated with the help of fuzzy systems. Some applications of fuzzy systems are Information Retrieval systems, Navigation systems, Robot vision etc.

Definition of Fuzzy logic: Fuzzy logic is derived from fuzzy set theory dealing with reasoning that is approximate rather than precisely deduced from classical ~~two~~ valued logic.

Note: A declarative sentence that is either true or false is a proposition.
Exceptions: Commands, Questions & Exclamations.

Rules of Inference

① Modus Ponens: $(P \wedge (P \rightarrow Q)) \rightarrow Q$

② Modus Tolens: $(\sim Q \wedge (P \rightarrow Q)) \rightarrow \sim P$

& Make Truth table for both

In fuzzy logic, the truth values are multivalued. such as Absolute true, partially true, Absolute false etc.

Fuzzy Proposition

A fuzzy proposition is a statement $\tilde{P}$ which acquires a fuzzy truth value, $T(\tilde{P})$

$\tilde{P}$: Ram is honest.

$T(\tilde{P})$: 0.8 (If $T(P) = 1$ ⇒ absolute honesty)

Fuzzy Connectives

1) Negation ⟶ $1 - T(\tilde{P})$

2) Disjunction ⟶ $\max [T(\tilde{P}), T(\tilde{Q})]$ If have $\tilde{P}, \tilde{Q}$ → 2 fuzzy propositions

3) Conjunction ⟶ $\min [T(\tilde{P}), T(\tilde{Q})]$

4) Conditional (Implication) → $\tilde{P} \Rightarrow \tilde{Q}$

$\sim \tilde{P} \cup \tilde{Q}$

$\max [(1 - T(\tilde{P}), T(\tilde{P})]$ ← In terms of fuzzy

Eg: $\tilde{P}$: Manoj is efficient · $T(\tilde{P}) = 0.8$

$\tilde{Q}$: Ram is efficient · $T(\tilde{Q}) = 0.65$

Calculate Negation of $\tilde{P}$. ⇒ $1 - T(\tilde{P}) = 1 - 0.8 = 0.2$

Calculate $\tilde{P} \cap \tilde{Q}$ ⇒ $\min [0.8, 0.65] = 0.65$

$\tilde{P} \cup \tilde{Q} \Rightarrow 0.8$

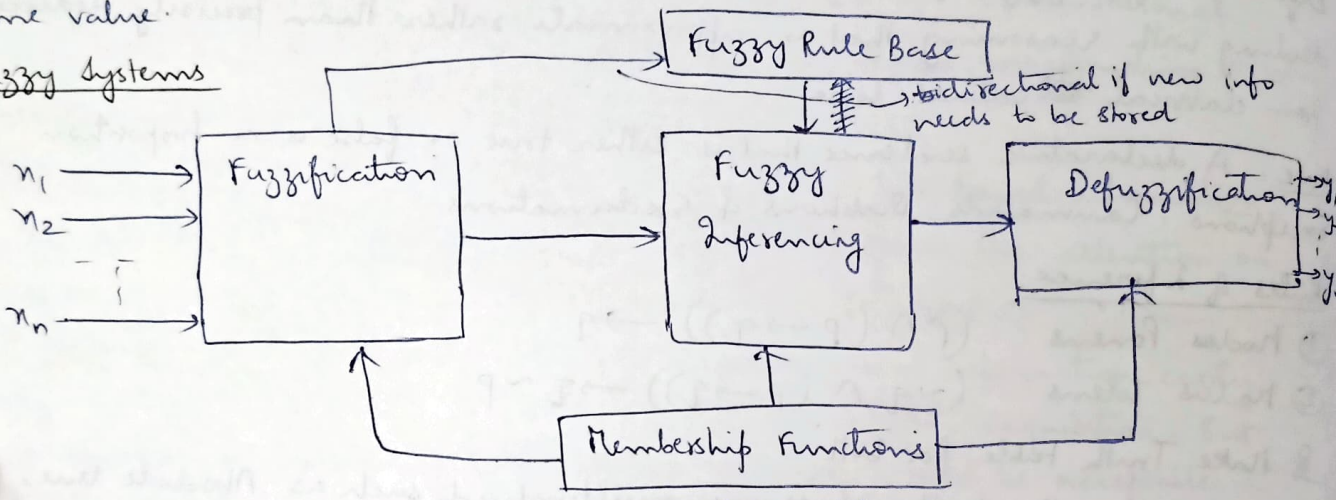
$\tilde{P} \Rightarrow \tilde{Q} \Rightarrow 0.65$

- **Fuzzy Quantifiers**: In crisp logic, the predicates are quantified by quantifiers → ordinary set in classical set/logic
~~with the use of which you ca~~ Similarly in fuzzy logic, the propositions are quantified by Quantifiers.

There are 2 classes of fuzzy quantifiers:

1) Absolute Quantifiers (around) (much greater than) - specify with some value.
2) Relative Quantifiers (almost) (about, most, atleast)

- **Fuzzy Systems**



Some properties which will be utilized

- **Fuzzy System Elements**

1) Input Vector: These are the crisp values which are transformed into fuzzy sets in the fuzzification block. It is represented by

$$X = [x_1, x_2, - - , x_n]^T \quad \text{(as represented by vectors)}$$

2) Output Vector: It comes out from the defuzzification block which transform an output fuzzy set back to crisp value.

3) Fuzzification: It is a process of transforming crisp value into grades of membership for linguistic terms such as Far, Small, Near of fuzzy sets.

4) Fuzzy Rule Base: It is a collection of propositions containing linguistic variables.

IF ... THEN

Eg: IF (x is A) (y is B) THEN (z is C). Here A, B, C are linguistic terms/variables

Variables

5) Membership Function: It provides a measure of the degree of similarity of elements in the universe of discourse (U) to fuzzy set.

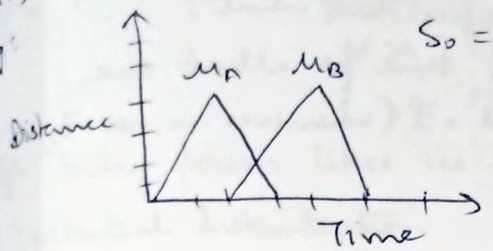
6) Fuzzy Inferencing: It combines the facts obtained from the fuzzification with

the rule based & conducts the fuzzy reasoning process.

7) Defuzzification: It translates the results back to the real world values. In many situations for a system whose output is fuzzy it is easier to take a crisp decision if the output is represented as a single quantity. The typical defuzzification methods are:

(i) Centroid Method
(ii) Centre of Sums
(iii) Mean of Maxima

eg:



S_0 = n_0/speed initial = 70 km/hr

$\mu_A = 0.75$

$\mu_B = 0.25$

17/1/22 Q Write short notes on knowledge based agents and read about example on wumpus world – Russel Navrig

→ First Order Predicate logic (FOPL) (or first order logic (FOL)) we connect it with well formed logic is another way of knowledge representation in AI. It's an extension to propositional knowledge. It is A powerful language that develops info about objects in a more easy way and also express relation between those objects. FOL talks about 4 things:

→ Syntax of FOPL: (i) Facts (ii) Objects involved in Fact (iii) Relationship b/w those obj (iv) Function to create for the relation to be used interoperably among facts

Some basic elements are constants (1, 2, Ryd), variables (x, y), predicates (brother, >, =), functions ($\sqrt{}$, $\sin$), connectives ($\rightarrow, \leftrightarrow, \vee, \wedge, \neg$), equality ($x = 3, y = 4$), quantifiers (universal and existing) (universal $\forall$, existential $\exists$)

→ Atomic Sentences

Most basic sentences of FOL and are formed from a predicate symbol followed by ~~predicate~~ parenthesis with a sequence of terms and represented as

Predicate (term 1, term 2, ... term n)

eg: Hari and Raghu are brothers. ⟹ brother (Hari, Raghu),

Tommy is a dog ⟹ dog (Tommy)

→ Complex Sentences

Made by combining atomic sentences using connectives. FOL statements divided in 2 parts.

(i) Subject main part of statement eg: x is an integer (x = Subject, is an integer = Predicate)

(ii) Predicate relation which binds atoms together in a statement.

→ Quantifiers

Language which generates quantification. They are Symbols that permit to determine the identity of variable in terms of range and scope in the logic expression.

Quantifiers

Universal (Range) — [for all, for every, for each]

Existential (Scope) — [for some, there exists, atleast one]

Universal Quantifiers:

Symbol of logical representation which specifies that statement within its range is true for everything and every instance of particular thing represented as $\forall$ (whenever we use $\forall$, we use $\Rightarrow$)

Existential Quantifiers: It is a type of quantifier ~~that~~ which expresses that the statement within its scope is true for atleast one instance of something. ~~It~~ It is represented as $\exists$ (whenever we use $\exists$, we use $\wedge$)

Eg: All men drink coffee

x^* variable

x_1 coffee $\wedge$ x_2 coffee

$\forall x$, man $(x) \rightarrow$ drink $(x,$ coffee$)$

(drink $(x,$ coffee$)$ — atomic; whole sentence — complex)

Eg: All birds fly ~~(forest)~~, $\forall x$ bird $(x) \rightarrow$ fly (x) → predicate: fly (bird)

Eg: Every man respects his parents $\Rightarrow$ respects (x, y) (parent)

$\forall x$ man $(x) \rightarrow$ respects (x, y)

$\forall x$ man $(x) \rightarrow$ respect $(x,$ parents$)$

$\forall y$ man $(x) \rightarrow$ respects (x, y)

Eg. Some boys are intelligent.

$x \rightarrow$ variable

x_1 intelligent $\vee$ x_2 intelligent

$\exists x$: boy $(x) \wedge$ intelligent (x)

→ Inferences in FOL

Used to deduce new facts and sentence from existing one

→ Terminologies

① Substitution → fundamental operation performed on terms and formulas.
- occurs on all inference systems in FOL.
- substitute a (const) for x (var), replace x by a. $F[a/x]$

② Equality: hard clause

Eg: brother (john) = smith

$\sim x = y \Rightarrow x \neq y$

→ Inference Rules for Quantifiers

① Universal Generalisation

It is a valid inference Rule which states that if $P(c)$ is true for any arbitrary element c in the universe of discourse then $\forall x\ P(x)$ is the conclusion.

② Universal Instantiation

- It is also called Universal Elimination.
- It can be applied to multiple times to add new sentence.
- We can infer any sentence $P(c)$ by substituting a ground term c $\forall x\ P(x)$ for any object in the universe of discourse.

$$\forall x\ P(x)$$
$$\therefore\ , c \in P(c)$$

Eg: 'Every person likes ice cream' then 'Ram likes ice cream'

③ Existential Instantiation

- It is also called existential elimination.
- A valid inference rule in FOL it can be only applied once to replace the existential sentences.
- One can infer $P(c)$ from the formula in the form

$\exists x\ P(x)$ for a new constant c.

$$\therefore\ , P(c)$$

④ Existential Generalization.

- It is also known as Existential Introduction.
- If there is some element $c \in U$ which has property P then we can infer that $\exists$ something in U that has property P

$$P(c)$$
$$\therefore\ , \exists x\ P(x)$$

→ Unification

It is all about making the expressions look identical.

- $P(x, F(y))$ and $P(y, F(z))$ } These 2 are identical if $x = y$ & $y = z$
- y/x and y/z

→ Conditions for Unification

1) Predicate symbols must be same, atomic sentences / expression with different predicate symbols can never be unified.
2) No. of arguments in both expressions must be identical.
3) Unification will fail if there are 2 similar variables present in same expression.

→ Unification Algorithm (Imp for exam)

Unify (L_1, L_2):

1) If L_1 or L_2 is a variable / constant then:

a) If L_1 and L_2 are identical then:

return NIL;

b) else if L_1 is a variable then:

(i) if L_1 occurs in L_2 then:

return FAIL;

(ii) else:

return (L_2/L_1) ;

c) else if L_2 is a variable then:

(i) if L_2 occurs in L_1 then:

return FAIL;

(ii) else

return (L_1/L_2)

d) else:

return FAIL;

2) If initiate predicate symbol in L_1 and L_2 are not identical then:

return FAIL;

3) if L_1 & L_2 have different number of arguments then:

return FAIL;

4) Set SUBST to NIL;

5) loop for i ← 1 to number of arguments in L1

a) Call UNIFY with i^{th} arg of L1 & L2 in S.

b) if S == FAIL then:

return FAIL;

c) if S! = NIL then:

(i) apply s to remainder of both L_1 & L_2;

(ii) SUBST = APPEND (S, SUBST);

6) return SUBST;

→ Implementation of Unification Algorithm

1) Initialize the SUBST = empty ;

2) Recursively unify atomic sentences ;

(i) Check for identical expression match ;

(ii) If one expression is a variable V_i and other is a term t_i which

doesn't has V_i then:

a) Substitute t_i/V_i in existing substitutions.
b) Add t_i/V_i to the SUBST list
c) If both the expressions are functions then:
functions names must be similar & no of arguments must be same in both expressions.

eg: $P(x, g(x))$

① $P(z, y)$, Is unification possible? => Yes, z/x, $y/g(x)$
② $P(z, g(z))$, Is unification possible? => Yes, z/x
③ P(prime, f(prime)) is not possible, f cant be replaced by F.

→ Predicate logic and Resolutions:

• Resolution in FOL:

- Resolution is a theorem proving technique that proves by contradiction.
- It is used when there are various statements given and there is a need to prove conclusions of those statement.
- Unification is a key concept in proof by resolution.
- Resolution is a single inference rule which can efficiently operate on CNF or Clausal form.
↳ Conjuctive Normal form.

→ CNF

- A sentence that is represented using conjuction of clauses.
- representation of OR's by vistue of AND's
- eliminate all implication and rewrite

Clausal form → disjuction of literals is called clause

→ Steps for Resolution

1) Conversion of facts into FOL.
2) Convert FOL into CNF
3) Negate the statements which need to be proved by Contradiction
4) Draw a Resolution graph.

eg: 1) 'If it is sunny and warm, you will enjoy'.
2) 'If it is raining you will get wet'
3) 'It is a warm day'
4) 'It is raing'
5) 'It is sunny'

To prove: 'You will enjoy'

Proof : Step1

① Sunny ∧ warm → enjoy

② raining → wet

③ warm

④ raining

⑤ sunny

Step 2 :

① ~(Sunny ∧ warm) ∨ enjoy

~ sunny ∨ ~warm ∨ enjoy

② (~ raining) ∨ wet

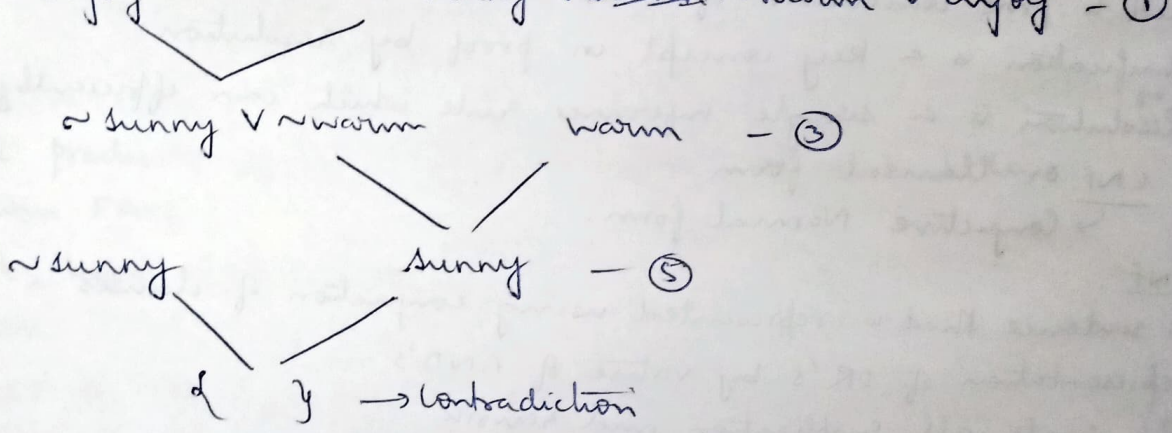
③ warm

④ raining

⑤ sunny

Step 3 : ~ enjoy

Step 4 : ~ enjoy ~sunny ∨ ~warm ∨ enjoy – ①



∴ we reach { } we can say we will enjoy.

eg: ① John likes all kinds of food

② Apple and vegetable are food

③ Anything and anyone eats and not killed is food

④ Anil eats peanuts and is alive.

⑤ Harry eats everything Anil eats

⑥ John likes peanuts – To prove

⑦ If not killed then alive

⑧ If alive then not killed.

Step1 : ① $\forall x$: Food(x) → likes (John, x)

② Food (Apple) ∧ Food (Vegetable)

③ $\forall x$, $\forall y$: eats (x, y) ∧ not killed(x) → Food(y)

④ eats (Anil, peanuts) ∧ alive (Anil)

⑤ ∀x : eats (Anil, x) → eats (Harry, x)

⑥ ∀x : not killed (x) → alive (x)

⑦ ∀x : alive (x) → not killed (x)

⑧ ?p → likes (John, peanuts)

Step 2 a) ① ∀x : ~ Food (x) ∨ likes (John, x)

② Food (apple) ∧ Food (vegetable)

③ ∀x, ∀y : ~ (eats (x, y) ∧ not killed (x)) ∨ Food (y)

④ eats (Anil, peanuts) ∧ alive (Anil)

⑤ ∀x : ~ eats (Anil, x) ∨ eats (Harry, x)

⑥ ∀x : ~ alive (x) ∨ not killed (x)

⑦ ∀x : ~ not killed (x) ∨ alive (x)

Step 2 b) ① ∀x : ~ Food (x) ∨ likes (John, x)

② Food (apple) ∧ Food (vegetable)

③ ∀x, ∀y : ~ eats (x, y) ∨ killed (x) ∨ Food (y)

④ eats (Anil, peanuts) ∧ alive (Anil)

⑤ ∀x : ~ eat (Anil, x) ∨ eats (Harry, x)

⑥ ∀x : killed (x) ∨ alive (x)

⑦ ∀x : ~ alive (x) ∨ not killed (x)

Step 2 c) ① ∀x : ~ Food (x) ∨ likes (John, x)

② Food (Apple) ∧ Food (vegetable)

③ ∀y, ∀z : ~ eats (y, z) ∨ killed (y) ∨ Food (z)

④ eats (Anil, peanuts) ∧ alive (Anil)

⑤ ∀w : ~ eats (Anil, w) ∨ eats (Harry, w)

⑥ ∀g : killed (g) ∨ alive (g)

⑦ ∀k : alive (k) ∨ ~ killed (k)

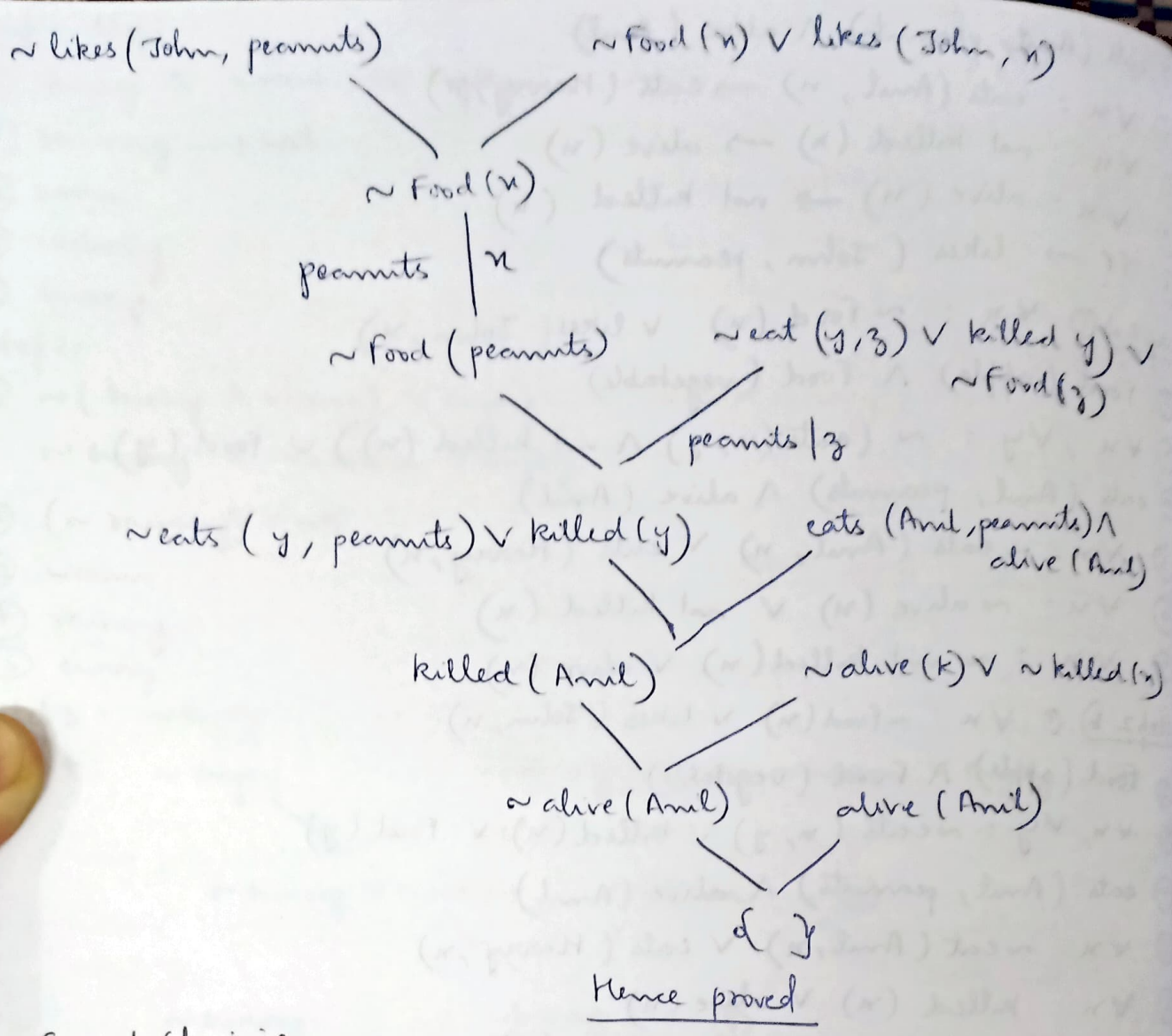
Step 2 d) Eliminate existential quantifier

Step 2 e) Drop Universal Quantifier

Step 2 f) Distribute conjuction over disjunction

Step 3 ~ likes (John, peanuts)

PTO



Hence proved

→ Forward Chaining

- Also known as forward reasoning.
- It is a form of reasoning which starts with atomic sentences in the knowledge base and applies inference rules in the forward direction to extract more data until a goal is reached.

- Properties

(i) It moves from bottom to top.

(ii) It is a process of making a conclusion based on known facts or data by starting from initial state and reach the goal state

(iii) Also known as data driven strategy as reaching to goal uses available data.

- Applications: expert systems

→ Backward Chaining

- It starts with the goal and works backward, chaining through rules to find known facts which support the goals.

- Properties

(i) It is a top-down approach
(ii) It is based on Modes Ponens inference rule.
(iii) Goal is broken into sub goals to prove that the facts are true.
(iv) It is a Goal driven approach as a list of goals decide which rules are selected and used.

→ Used in game theory, automatic theorem proving tools (resolutions) and inference engines, proofs assistant etc.

→ Difference b/w Forward and Backward Reasoning

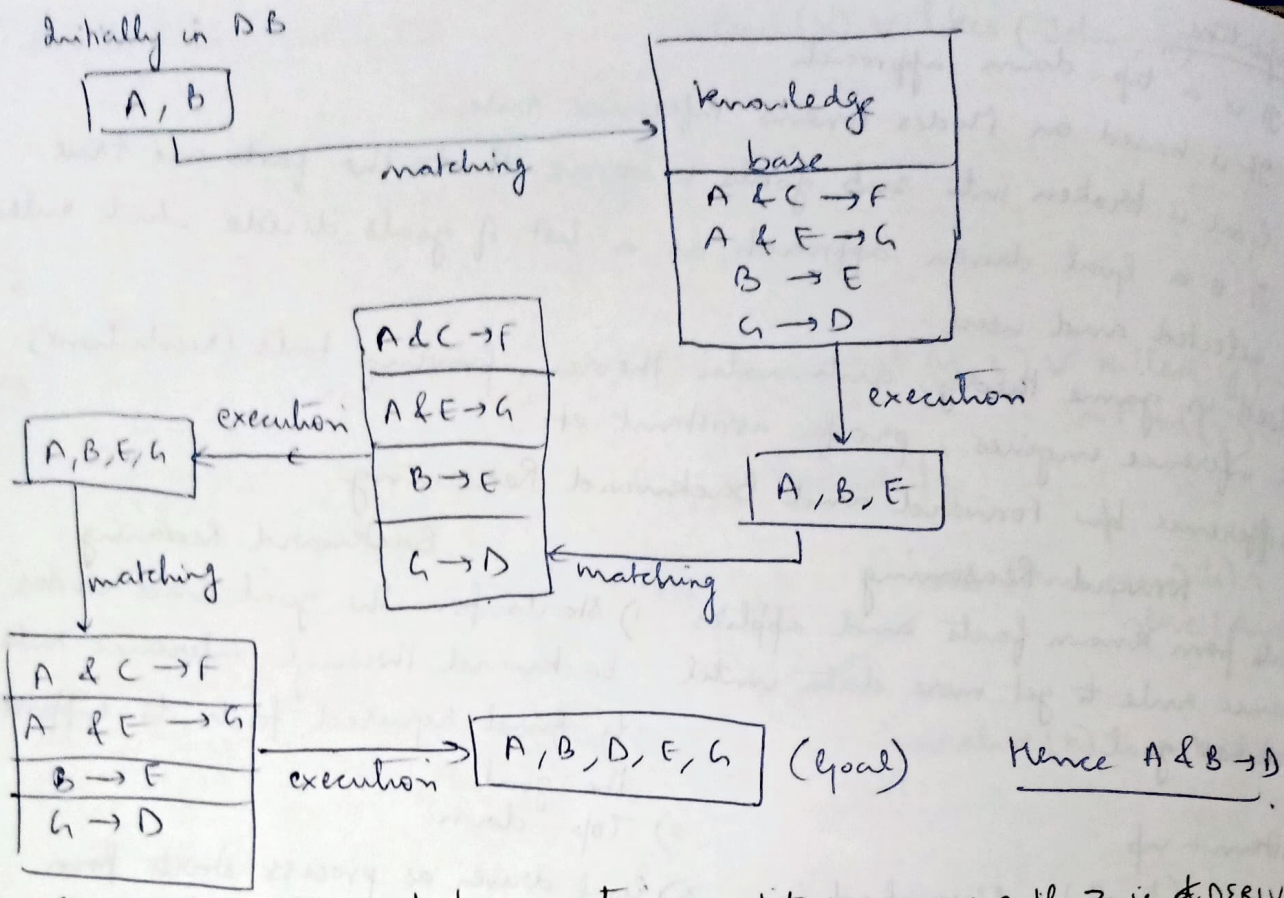
Forward Reasoning	Backward Reasoning
1) Starts from known facts and applies inference rule to get more data until it reaches goal.	1) Starts from the goal and works backward through inference rule to find required facts to support the goal.
2) Bottom-up	2) Top-down
3) Data driven as goal reached using available data	3) Goal driven as process starts from goal and divided into sub goals to get facts.
4) Uses bfs strategy	4) Uses dfs strategy
5) Tests for all available rules	5) Only tests for few required rules.
6) Suitable for planning, monitoring, controlling & interpretation application	6) Suitable for diagnostic, descriptions & debugging applications
7) Can generate infinite no. of possible conclusion.	7) Finite no. of conclusions can be generated.
8) Aimed for any conclusion	8) Only aimed for required data
9) Works in forward direction	9) Works in backward direction

→ Rules:

Eg: ① IF A AND C THEN F
② IF A and E THEN G
③ IF B THEN E
④ IF G THEN D

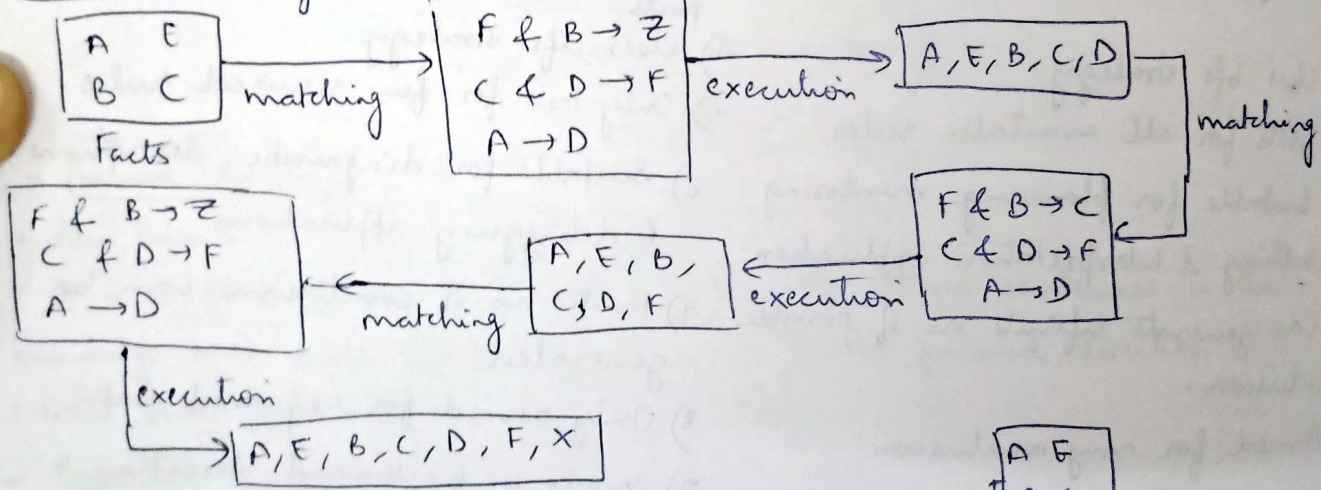
Prove that if A and B are true then D is True.

PTO

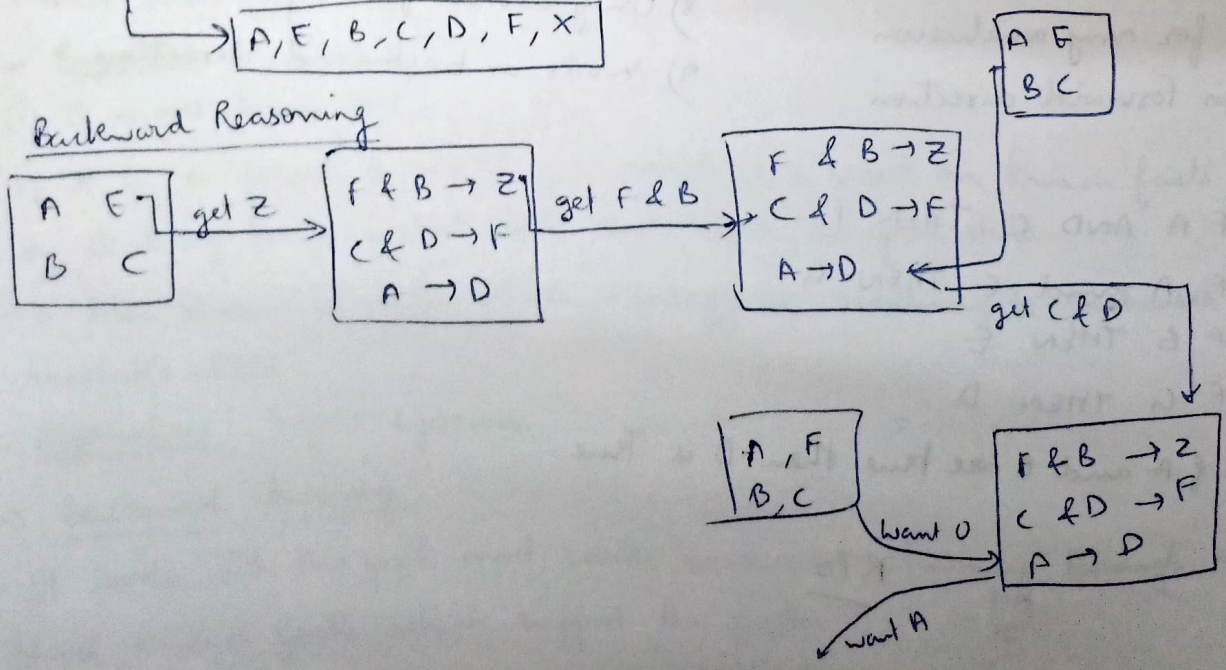


Eg. Goal state is Z and termination condition is STOP if Z is ~~&~~ DERIVED

Forward Reasoning



Backward Reasoning



3/10/22

Horn Clause → A clause is called Horn clause if it contains atmost 1 positive literal usually written as:

$$L_1, L_2, \dots, L_n \Rightarrow L \quad (\equiv \neg L_1 \vee \dots \vee \neg L_n \vee L) \quad \text{(+ve Skolemization)}$$

where $n \geq 0$

2. A disjunction of literals with atmost one negative literal is known as Dual Horn Clause.
3. Horn clause with exactly 1 positive literal → Definite Clause / Strict Horn Clause.
4. A define clause with no negative literal → Unit Clause
5. A unit clause without variables → Fact
6. Horn clause without positive literal → Goal Clause

→ **Knowledge Base (KB)**

1. KB is a set of sentences in a formal language.
   ↳ Lang. that will be understood by all
2. Syntax is the sentence of the language. & Semantics is the meaning of the sentence.

Syntax, Semantics } part of possible world

→ **Model**

It is a mathematical abstractions each of which simply fixes the truth ~~and~~ or falsehood of every relavant sentence.

Eg: In possible world, $x \rightarrow$ boy, $y \rightarrow$ girl $x + y = 4 \Rightarrow$ There are 4 people in the world

In model, assign some real values to x & y $\Rightarrow$ $x \rightarrow R$, $y \rightarrow R$

If a sentence α is true in model m, we say that m satisfies α or m is a ~~root~~ model of α. $M(\alpha) \rightarrow$ Set of models of α.

→ **Entailment**

A sentence follows logically from another sentence & represented as:

$$\alpha \models \beta \quad , \quad \alpha, \beta \text{ are sentences}$$

↳ entails

Formally α entails β if and only if in every model where α is true, β is also true. $\Rightarrow M(\alpha) \subseteq M(\beta)$

↳ relation established

→ **Model Checking**

- It enumerates all possible models to check that α is true in all models in which KB is true. represented by $M(KB) \subseteq M(\alpha)$
- If an inference algorithm i can derive α from KB, then $KB \vdash_i \alpha$ (α is derived from KB using i or i derives α from KB) ← means

- Soundness

Algorithm i is sound if whenever $KB \vdash_i \alpha$ is true, It is also true that $KB \models \alpha$.

- Completeness

i is complete if whenever $KB \models \alpha$ it is also true that α is derived from KB using i or $KB \vdash_i \alpha$. Thus the procedure of model checking will answer any question whose answer follows from what is known by KB.

- Validity

A sentence is valid if it is true in all models. ($A \wedge \sim A \rightarrow F$ in every universe)

- Validity is connected to inference via deduction Theorem. It says that $KB \models \alpha$ is true or valid if and only if $KB \Rightarrow \alpha$ is valid.

$$KB \Rightarrow \alpha \equiv \sim KB \vee \alpha$$

- A sentence is satisfiable if it is true in some model. $A \vee B$ (Both should be valid)
- A sentence is unsatisfiable if it is true in no models.

eg: $A \wedge \sim A$ will never be true.

- Satisfiability is connected to inference as $KB \models \alpha$ if and only if $(KB \wedge \sim \alpha)$ is unsatisfiable.

(both should not be true & false at same time)

2/11/22

UNIT-4

① Planning - It is a process of computing several steps of problem solving before executing any one of them. It consists of:

a) Choose the best rule i.e Heuristics
b) Apply the rule on the problem to get new state or problem
c) Detect if solution is found and stop
d) Detect dead end so that redirections are explored.

Q How to choose the rule?

① Consider the goal state and compare with initial state.
② Identify the relavant rules to be applied heuristically.

- State - space search

It is used in problem solving and is a process used in AI in which successive configurations or states of an instance are considered with an intention of finding the goal state with desired property. The problems are modelled as state space which is a set of states in which a problem can be