16/10/24

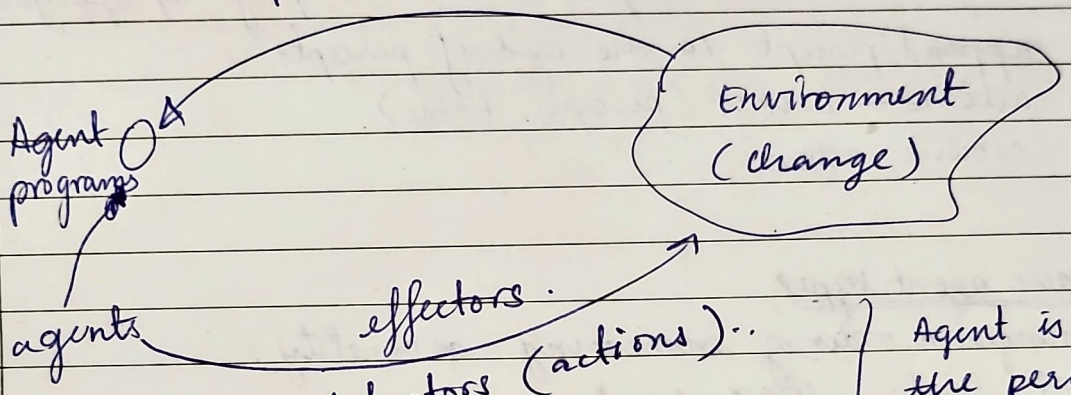
Date :

UNII - 1 Chapter 1: Intro

Chapter 2 : Agents

eg: human agent, software agent (chatbot), robotic agent (robotic vaccum cleaner)

Agent = Architecture (H/W) + program (function)



Agent is anything that perceives its environment through sensors and acting upon the environment through actuators

Good agent → high performance
→ optimized result
→ Rational action.

Factors of rationality:
- P → performance (how measured)
- E → Environment (interaction with)
- A → Actions (output)
- S → Sensors (input)

* Rationality → do the right thing
  - ↳ performance measure.
  - ↳ prior knowledge of environ.
  - ↳ actions
  - ↳ percept sequence.

  → maximize expected performance.

* Omniscience → having unlimited knowledge (both and right thing)
  - ↳ actual performance.

* Environment → problem rational agents → solution

Date :

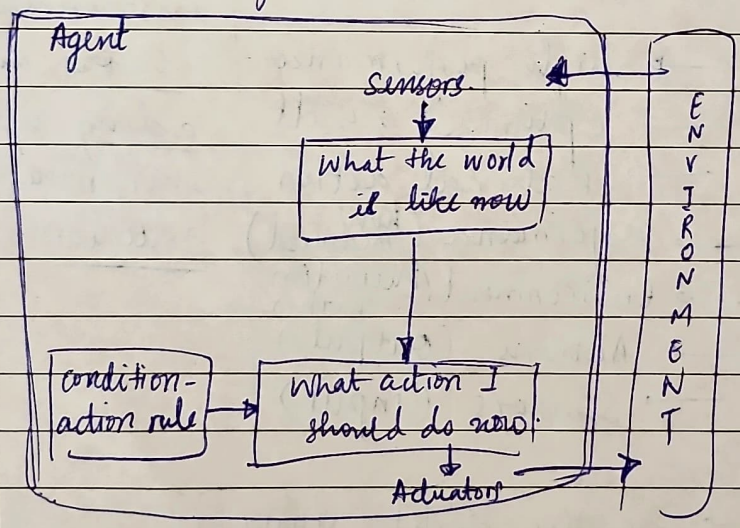
Agent program

function TABLE-DRIVEN-AGENT (percept) returns an action
  persistent: percepts, a sequence initially empty
    table, a table of actions indexed by percept sequences initially fully specified
  append percept to the end of percepts
  action ← LOOKUP (percepts, table)
  return action.

Basic agent types

Arranged in order of increasing generality:

① Simple reflex agent



function SIMPLE-REFLEX-AGENT (percept) returns an action
  persistent: rules, a set of condition-action rules

  state ← INTERPRET-INPUT (percept)
  rule ← RULE-MATCH (state, rules)
  action ← rule.ACTION
  return action

AI

- Acting humanly: Turing Test approach.
- Thinking humanly: Cognitive modelling approach.
- Thinking rationally: Laws of thoughts approach.
- Acting rationally: Rational agent approach.

Foundations of AI

- Mathematics
- Philosophy
- Cognitive science
- Psychology
- Linguistics
- Biology
- Economics
- CSE

List the foundations → 2 marks. Explain - 4/6 marks.

Difference b/w informed and uninformed/search strategies.

Informed	Uninformed / blind / brute force
* It searches with info.	* W/o info.
* uses knowledge based sol^n.	* no knowledge is used.
* gives quick sol^n	* time consuming
* less complexity (in terms of time & space)	* more complexity
* A*, heuristic, best first search	* BFS, DFS → LIFO * FIFO

Problem Formulation

1. Initial state
2. Possible actions
3. Transition model / Successor function
4. Goal Test
5. Path cost

Measuring problem-solving performance

* Completeness
* Optimality
* Novelty
* Time complexity
* Space complexity

① Uninformed search

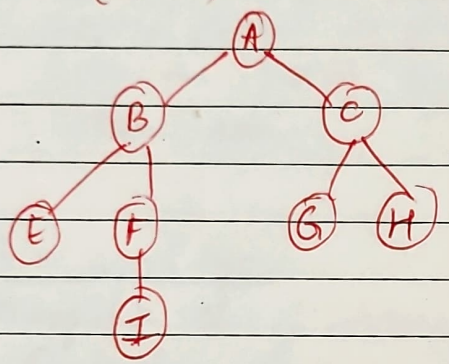
↳ ① BFS
② Uniform Cost search (UCS)
③ Depth first search (DFS)
④ Depth limited search Algorithm
⑤ Iterative deepening DFS
⑥ Bidirectional search

23/10/24

Date :

1 BFS

* Queues (FIFO)



Level 0
Level 1
Level 2
Level 3

Drawback

* More memory and time consumption (goes to every node)

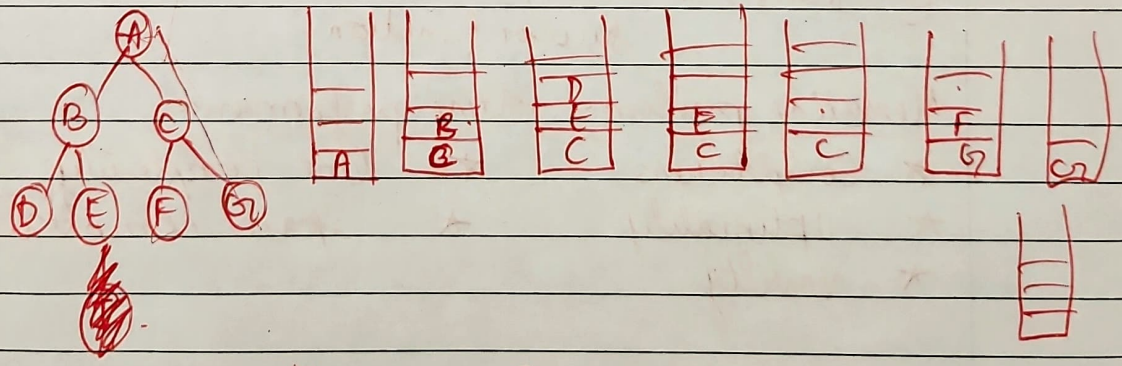
optimal eg: A → I.

~~A → B → E → F → I~~ ∅ A → B → C → E → F → G → H → I

2 UCS

* Based on ~~cd~~ cost on edges, choose shortest path to go to destination.

3. DFS (LIFO)



A → B → D → E → C → F → G.

* Can go to infinite search.; depends on what node is inserted.
* no guarantee of finding solution

4] DLS (not optimal) * memory efficient

* We set a limit of level. till (till what depth to go)
level

Drawback: If the limit doesn't contain the goal.

5] ID DFS. (increment the limit)

Iterative check for each limit till goal is found

6] Bidirectional

* At a particular time, graph divided into and simultaneously checked till they reach a common node.
* search stops when graphs intersect (for sure)
(same space and time complexity).
* fast.

Informed search

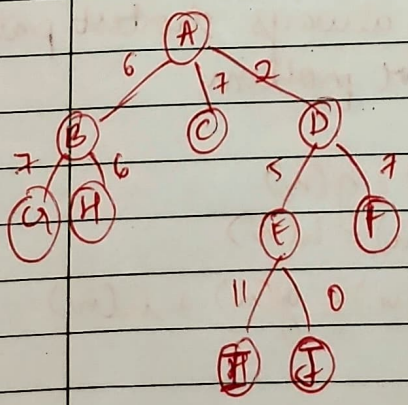
* uses idea of heuristic
  - Best first search
    - greedy best first search.
    - A* search
  - heuristic search.

search strategy < evaluation function
estimate of desirability

h(n) → heuristic function: cost from source to goal node.
(most promising path)

1] Best first search → BFS + DFS.

* Maintains two list: OPEN (ascending) & CLOSED (descending)



open list

A
2 6 7
D B C
6 7 7
E B C F
0 6 7 7 11
J B C F I.

closed list

B

A D E J.

* It can get stuck in loops * Not optimal
* $O(b^m)$ → depth, branching factor * Destination h(n) will be zero

* not optimal as uses estimated cost not actual cost
* Better than BFS and DFS.

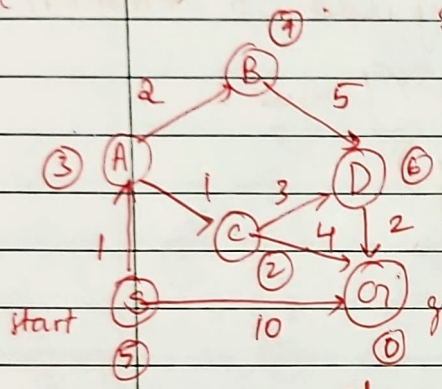
A* search

(12 marks)

$f(n) = g(n) + h(n)$

$g(n)$: cost from start to n

$h(n)$: cost from n → goal



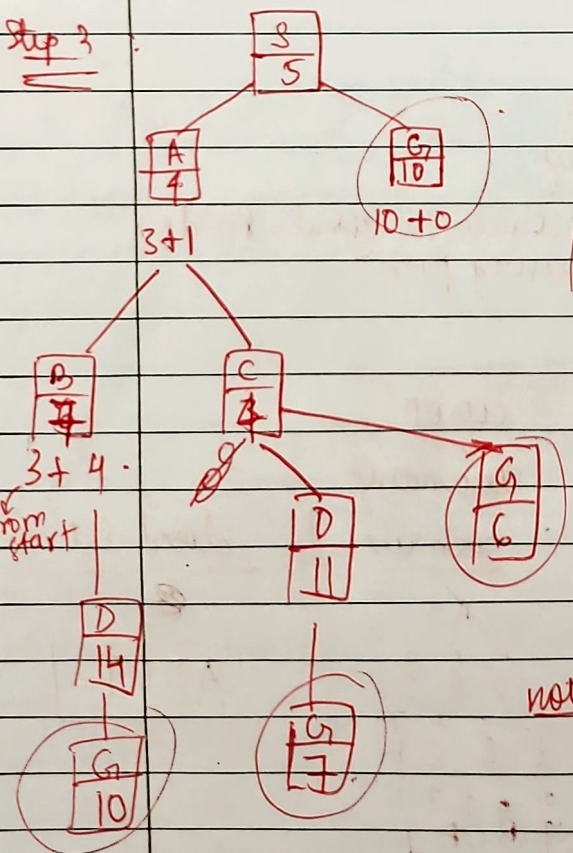
start state = S
goal state = G

Step 1:

$f(n) = g(n) + h(n)$

S
5
0+5

Step 2:



optimal:

S → A → C → G

* best algorithm
* optimal and complete
* not always shortest path.
* solve problem

note: UCS : $f(n) = g(n)$
Best FS : $f(n) = h(n)$
A^* : $f(n) = g(n) + h(n)$.

UNIT - II

Date : 30/10/24

Machine → valuable

3 factors { knowledge, reasoning, intelligence } → info about env. → decision.

(understand)

Knowledge based agents → use task specific knowledge

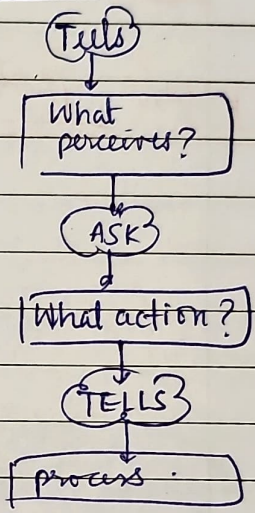
- knowledge base (KB) (store in sentences)
- inference system/engine (get new sentences)
- TELL, ASK
- * domain independent algorithm.
- * domain specific content
- * central component

knowledge base

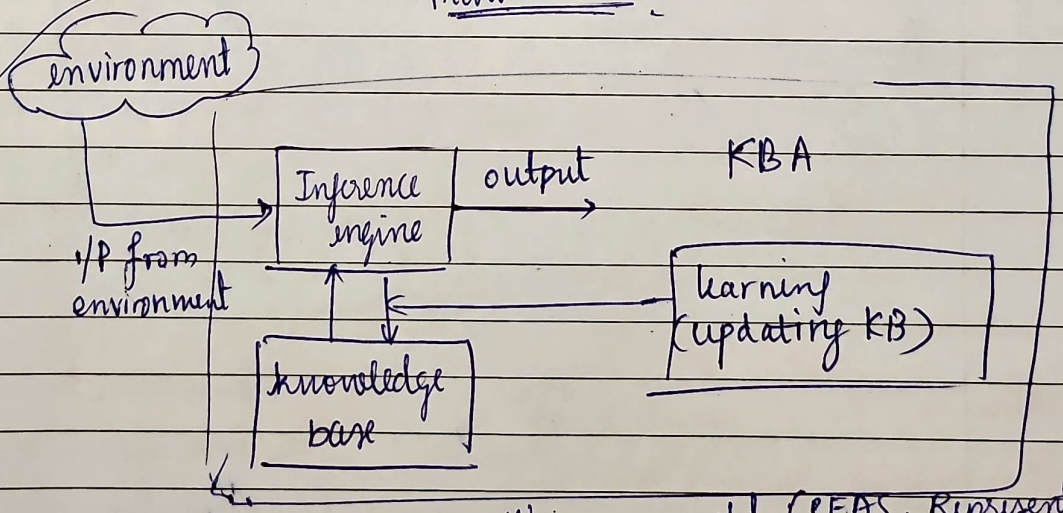
ASK — query what is known from KB

TELL — add new sentence to KB

Generic KBA



Architecture



Approaches

- declarative — eg: constraints, rules
- procedural — eg: functions

* Wumpus world (PEAS, Representation)
* logic in general
  - ↳ syntax – sentences in language
  - ↳ semantics – meaning of sentences (true or false)
    - ↳ model – variable states takes ~~a pair~~ specific value.

eg: 2 inputs will have 4 models

Date: 6/11/24

Propositional logic

∴ No. of 2^n models created for n inputs.

Notation
- α (sentence)
- m (model)
  - m satisfies α / m is a model of α (if α is true)
  - $M(\alpha)$ - set of all models for α (true ones only)

* Entailment - relationship/mapping of 2 sentences
  $\alpha \models \beta$: α entails β. i.e. If β is true, α is also true

note: $KB \models \alpha$ iff $M(KB) \subset M(\alpha)$.

* Inference. $\boxed{KB \vdash_i \alpha}$ if i can derive α from KB (i creates entailed sentences).
* Sound / TRUTH-PRESERVING. (i creates entailed sentences)
* Proportional Logic
  - Atomic (one symbol)
  - Complex (combo of sentences)

① ¬ ② ∧ ③ ∨ ④ → ⑤ ↔

* BNF notation

Sentence ⟶ AtomicSentence | ComplexSentence
AtomicSentence ⟶ True | False | P | Q | R | . . .
ComplexSentence ⟶ (Sentence) | [Sentence]
| ¬ | ∧ | ∨ | → | ↔

order precedence: ¬, ∧, ∨, →, ↔

* Theorem proving
  - Logical equivalence
    $\alpha \equiv \beta$ if α & β are T in same m.
    ~~$\alpha \wedge P \wedge Q \equiv Q \wedge P$~~
    $\alpha \equiv \beta$ iff $\alpha \models \beta$ and $\beta \models \alpha$.
  - Validity
    α is valid if it is T for all models.
    * valid sentences ⟶ tautology.
    $\alpha \models \beta$ iff $\alpha \Rightarrow \beta$ is valid
  - Satisfiability
    α is satisfiable if it T for some model
    * SAT problem.

7/11/24

Inference Rules:

① Modus Ponens

$$\frac{\alpha \to \beta \quad \alpha}{\therefore \beta}$$

② And-elimination

$$\frac{\alpha \wedge \beta}{\therefore \alpha}$$

③ Resolution

= propositional / first order predicate

$$\frac{(\alpha \vee \neg\beta) \wedge (\gamma \vee \beta)}{\therefore \alpha \vee \gamma}$$

only → *clause

CNF sentence → clause$_1 \wedge \dots \wedge$ clause$_n$

clause → Literal$_1 \vee \dots \vee$ Literal$_m$

Literal → Symbol | $\neg$Symbol

Symbol → P | Q | R | ....

step ②: $\therefore \beta$

note: $\alpha \to \beta \iff \equiv \neg\alpha \vee \beta$

* Monotonicity

if $KB \models \alpha$ then $KB \wedge \beta \models \alpha$

* Two Normal Form

step ①: CNF (Conjunction) ← DNF (Disjunction)

note: $\alpha \iff \beta \equiv (\alpha \to \beta) \wedge (\beta \to \alpha)$

step ③:

$\neg(\neg\alpha) \equiv \alpha$

$\neg(\alpha \wedge \beta) \equiv \neg\alpha \vee \neg\beta$

$\neg(\alpha \vee \beta) \equiv \neg\alpha \wedge \neg\beta$

step ④: Distributive & ($\Leftrightarrow$) Commutative

eg: $A \leftrightarrow (B \vee C)$

① $(A \to (B \vee C)) \wedge ((B \vee C) \to A)$

② $(\neg A \vee (B \vee C)) \wedge (\neg(B \vee C) \vee A)$

③ $(\neg A \vee (B \vee C)) \wedge ((\neg B \wedge \neg C) \vee A)$

④ $(\neg A \vee B \vee C) \wedge (A \vee \neg B) \wedge (A \vee \neg C)$

it is CNF.

Inference

Soundness — Completeness

*Horn clause — a disjunction with atmost one +ve literal

- definite: one positive
- mid: only one positive
- goal: no positive

12/11/24

** Forward and backward chaining.

Facts → Rules → Conclusion

→ data driven, bottom up approach.

Conclusion → Rules → Facts

goal driven, top down approach.

** First order logic

↳ predicate logic

(collection of objects, attributes, relations, functions.)

Statements — subject — Predicate

Shorthand — Predicate (subject)

eg: x is an integer → integer(x)

FOL — Syntax — which is logical

— semantics

Quantifiers — Universal ($\forall$) →

— Existential ($\exists$) → $\wedge$

↓

nested quantifiers

Syntax: (FOL)

Sentence → Atomic | Complex

Atomic → Predicate | Predicate(Term)

Complex → Sentence | ¬ | ∧ | ∨ | → | ⇔ | Quantifier variable ... Sentence

Terms → Function [Term] | Constant | variable.

Quantifier → ∀ | ∃

Constant → A | X_1 | John | ...

variable → a | x | s | ...

Predicate → True | False | After | Loves | Raining

Function → Mother | LeftLeg ...

Operator Precedence : ¬, =, ∧, ∨, =>, <=>

Date : _____

note: Sentences are added to KB using TELL.
Ask questions from KB using ASK.

note: We can use negation in quantifier also.

Domain — Family relationship (kinship domain)
Domain — Wumpus world.

Steps in knowledge engineering process: (10 marks).

1. Identify the task - figure out the range of questions that can be asked (based on PEAS)
2. Assemble relevant knowledge: - requirement analysis
3. Decide on vocabulary of predicates, func^n, constants
4. Encode general knowledge abt domain
5. Encode description of specific problem instance
6. Pose queries to inference procedure and get answers
7. Debug the knowledge base.

1. using PEAS
   - identify the next steps (functionality, structure)
2. related to problem, take model and analyse.
   - find the rules.
3. From statements
   - predicate
   - function
   - constants.
4. - Record axioms
   - specify the meaning
   - elaborate the issues.
5. convert general statements to FOL statements
6. ask queries and find solutions
7. testing process. (to find any issues).

	Propositional	vs	First order.
Ontology	Facts (P, Q).		objects, functions, relations
Syntax	Atomic connectives		var, quantifiers., terms.
Semantics	Truth Table		Interpretation
Inference Algo	DPLL, GSAT		Unification, Forward/Backward chaining
complexity	NP-complete		Semi decidable.

Date :

Inference :

① Substitution.

SUBST (θ, α)

↓ Sub. ↓ Sentence

$\{t1/v1, \ldots, tn/vn\}$

where ti – term

vi – variable

no two vi can be same.

{ f(z)/x , y/z }

replace x with f(z)

z with y

Inference rules for quantifiers

* Universal Generalization
* Universal Instantiation
* Existential Instantiation
* Existential Introduction / Generalization.

Instantiation / Elimination

$$\frac{P(c)}{\therefore \forall x\ P(x)}$$

$$\frac{\forall x\ P(x)}{\therefore P(c)}$$

$$\frac{\exists x P(x)}{\therefore P(c)}$$

$$\frac{P(c)}{\therefore \exists x\ P(x)}$$

Unification

eg: $p(x, F(y))$ —— ①

$p(a, F(g(z)))$ —— ②

① = ② $[a/x, g(z)/y]$ } subs. set.

$p(a, F(g(z))$ —①

$p(a, F(g(z))$ —②.

Conditions of unification:

* predicate symbol must be same.
* no. of args. must be same
* if two same var in same expression, unification will fail.

eg: $Q(a, g(x,a), f(y))$

$Q(a, g(f(b),a), x)$

$[f(b)/x]$

$Q(a, g(f(b),a), f(y))$

$Q(a, g(f(b),a), f(x))$

$[f(b)/x] \cdot [b/y]$

$Q(a, g(f(b),a), f(b))$

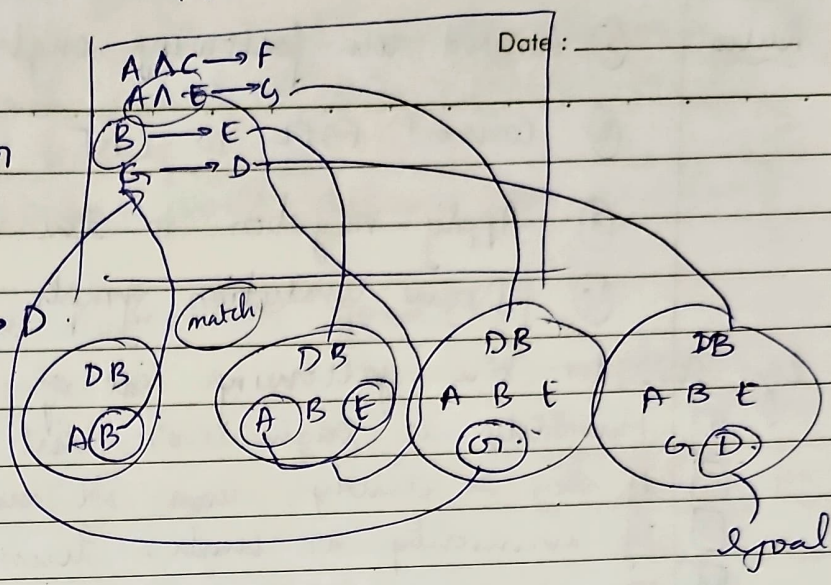
$Q(a, g(f(b),a), f(b))$

Forward chaining

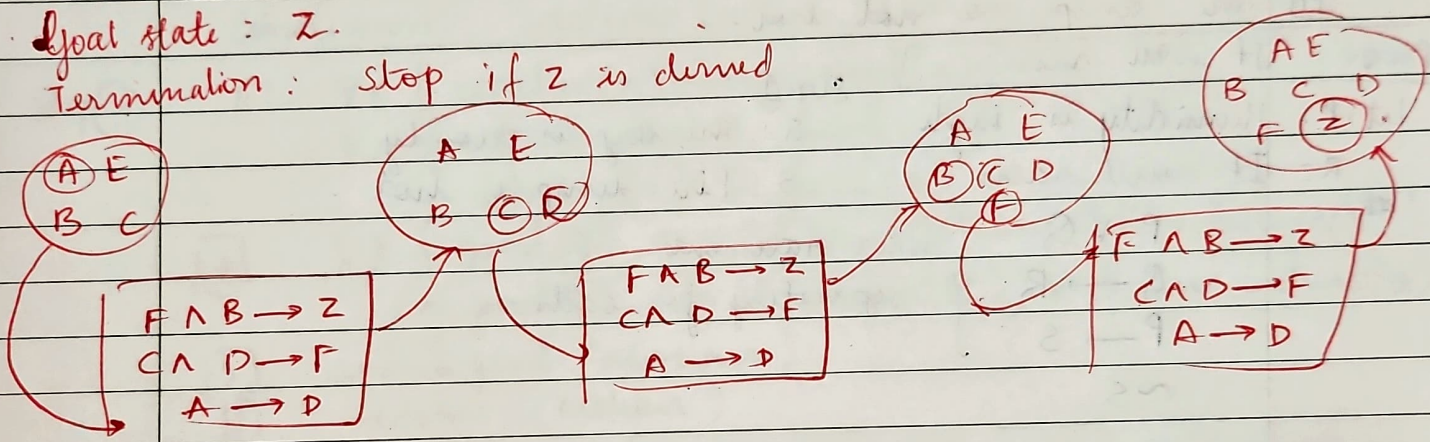
eg: Rule 1: If A & C then F
Rule 2: If A & E then G
Rule 3: If B then E
Rule 4: If G then D

Goal → D. ⟹ A ∧ B → D

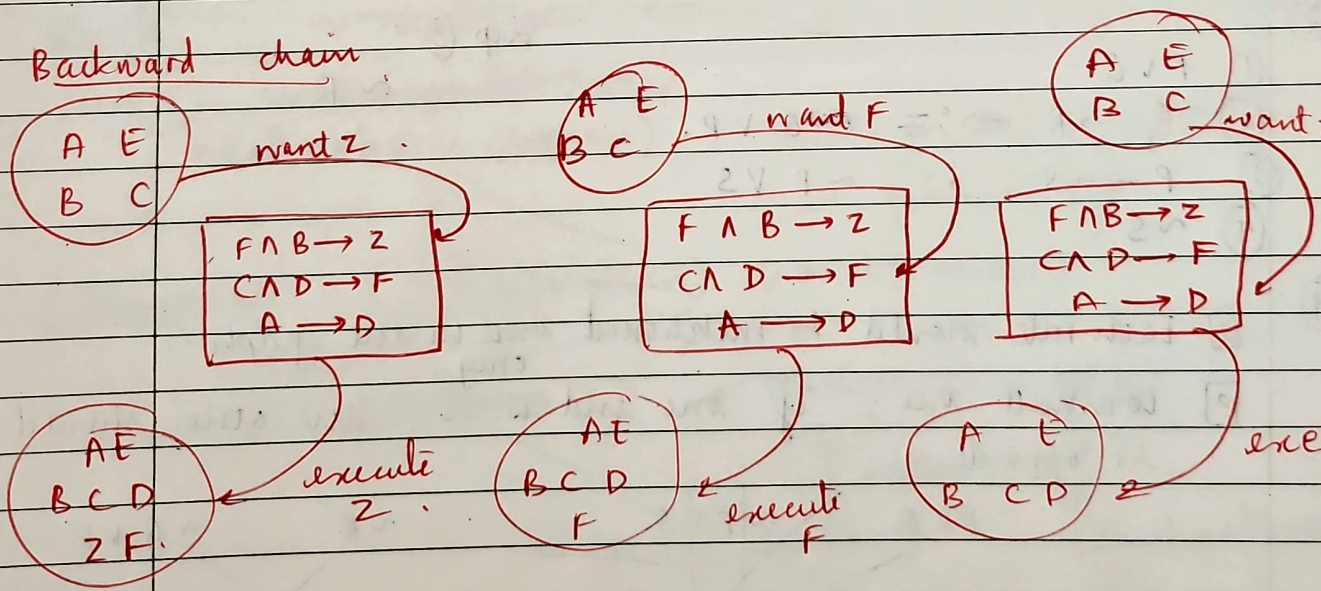
Initial → A, B (Database)



Goal state : Z.
Termination : Stop if Z is derived



Backward chain



Resolution

Rules:
1. Convert the following english statements into FOPL (First order Predicate Logic)
2. Convert FOPL to CNF.
3. Apply negation to the conclusion part
4. Draw resolution graph.

Eg: Consider the following set of facts:
1. Humidity is high or the sky is cloudy.
2. If sky is cloudy, then it will rain.
3. If humidity is high, Then the temp is hot.
4. The temp. is not hot.

Prove: It will rain.

Step 1:

Let P: Humidity is high Q: The sky is cloudy.
R: It will rain S: The temp is hot.

$P \vee Q$
$Q \rightarrow R$
$P \rightarrow S$
} write statements separately for each.

$\sim S$

Step 2:

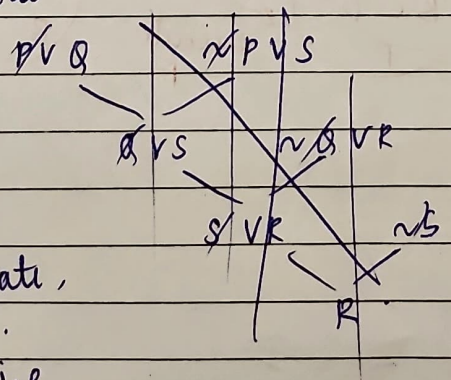
1. $P \vee Q$
2. $Q \rightarrow R \quad :\equiv \sim Q \vee R$
3. $P \rightarrow S \quad := \sim P \vee S$
4. $\sim S$

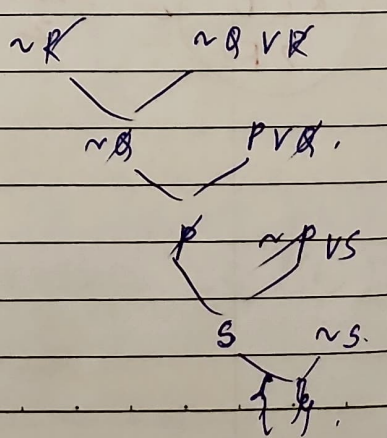
Step 3:

$\sim R$

Step 4:

1. Each rule should be mentioned once in the graph only.
2. Construct tree; if one end is $\sim$, the other should be opposite.





If true reaches null state, it means $\sim R$ is false. Hence R is true. i.e "It will rain."
Hence proved!

→ Hence proved.

~~Eg: Define different kinds of food.~~

Resolution: (1) All teachers are good (2) Anyone who is good and intelligent will deliver ~~decent~~ excellent lecture (3) Mohan is an intelligent teacher.

Show that Mohan will deliver an excellent lecture.

Step (1) :

~~Let P:~~ Let (1) $\forall x [teacher(x) \rightarrow good(x)]$.

(2) $\forall x [good(x) \wedge intelligent(x) \rightarrow deliverlecture(x)]$

(3) intelligent (Mohan) $\wedge$ teacher (Mohan).

Conclusion: deliver lecture (Mohan).

Take T(x) g(y) i(y) L(y).

Step (2) - (1) $\forall x [teacher(x) \rightarrow good(x)]$.

$\forall x [\sim teacher(x) \vee good(x)]$.

$\sim teacher(x) \vee good(x)$

(2) $\forall x [good(x) \wedge intelligent(x) \rightarrow deliverlecture(x)]$.

$(\sim good(x) \vee \sim intelligent(x)) \vee deliver\ lecture(x)$

(3) intelligent (Mohan).

(4) teacher (Mohan).

Step (3) ~~$\sim intelligent(Mohan)$~~ (5) $\sim$ deliver lecture (Mohan).

Step (4) ~~$\sim intelligent(Mohan)$~~

$\sim deliver\ lecture(Mohan)$. $\sim g(x) \vee \sim i(x) \vee L(x)$.

$\{Mohan/x\}$

$i(Mohan)$ $\sim g(Mohan) \vee \sim i(Mohan)$

$\{\}$

$\sim g(Mohan)$ $\sim teacher(x) \vee g(x)$

$\{Mohan/x\}$

$\sim teacher(Mohan)$ $teacher(Mohan)$

$\{\}$

$\square$

Hence proved.