Artificial Intelligence:- AI is the study of How to make computer do things which people do better. [machine + human Intelligence]

↳ AI Can Cause a machine to work as human.

↳ AI → Artificial [Man-Made]
→ Intelligence [Power of thinking]

GOALS OF AI: i) Replication of Human Intelligence.
ii) Solving problems – that require Knowledge.
iii) Building a machine that Can do human Intelligence task. [CHESS, Proving theorem, automated car driving ...]
iv) Providing advise to the User.
v) Intelligent Connⁿ b/w a perception and Connⁿ.

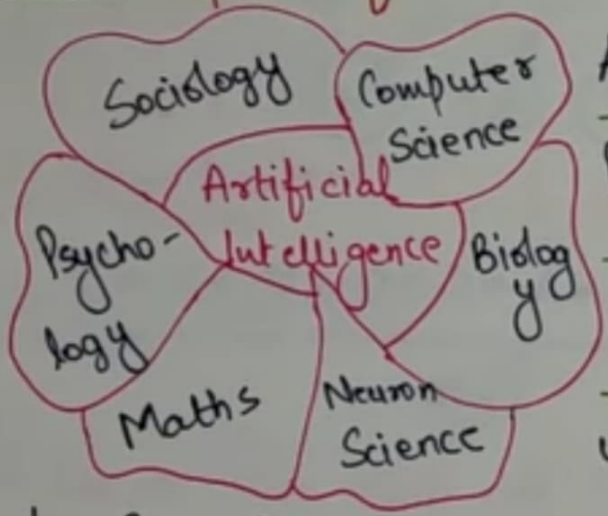
Reasons of Boost in AI:

↳ i) S/W or device Can be made to solve Real-time Problems.
ii) Creation of Virtual assistant [SIRI, CORTANA]
iii) Robots development. [Helps in dangerous env. condⁿ]
iv) New JOB opportunities.

Applications of AI:-

1. **AI in Gaming**: Chess, Poker, tic-toe.
   ↳ machine can think large no. of moves.
2. **AI in NLP**: Natural lang. Processing
   ↳ Machine can understand human lang.
3. **AI in HealthCare**: Fast diagonsis
   ↳ Robotic Surgery.
4. **AI in Finance**: Adaptive Intelligence.
   ↳ automatic chatbots, algorithm trading.
5. **AI in Data Security**: Helps in making data/appln more secure.
   ↳ AEG bot, AI2
6. **Expert System**: Integration of s/w, machine and special infon to provide reasoning & advise.
7. **Computer Vision**: Understand the visual automatically by machine.
8. **Speech Recognition**: Extract the meaning of sentence by human talk. [slang removal, noise rem.]
9. **Robotics**:
   Talk and behave like humans. ↳ Erica and sophia.
10. **AI in e-Commerce**: Automatic recommendation of product, Service requ.

AI is Comprised of :-



- Reasoning
- Learning
- Problem solving
- Language Understanding.

Advantages of AI	Disadvantages of AI
Accuracy ↑ & Error ↓	COST ↑
Fast Decision Making.	Can't thing beyond the limits.
Reliability is more	No feeling & emotions.
usefulness in Risky Area.	more dependency on machines ↑.
Digital Assistant	No original thinking.

Classification of AI:-

WEAK AI: (Narrow AI) Able to perform dedicated task with Intelligence. [Not concerned with HOW]
- ↳ Can't perform beyond its field or limitations.
- ↳ Example:-
  - Flying machine
  - Using logics
  - Apple SIRI
  - Playing chess

Evolutionary AI: It is the study and design of machines that simulate simple creatures and attempt to evolve.
- ↳ Example: Ants, Bees etc.

Strong AI: It is the study and design of machines that simulate human mind to perform intelligent tasks.
- ↳ i) Borrowing Ideas from psychology and neuroscience
- ↳ ii) Forgetting things, Genetics, Language.

Super AI:- Hypothetical Concept.
machine > Human.
[machine ⟷ machine] (Communication).

Types of AI Agents:- [Responsible for any work output obtained from System.

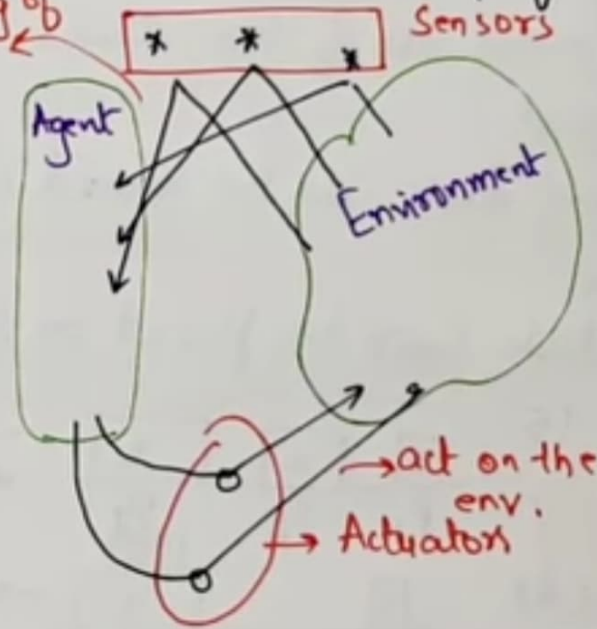
AI is defined as Study of Rational Agents.
- Person
- Firm
- Machine/s/w

Make Decisions

AI System is Composed of
- Agent
- Environment

Agent is anything like:-
- i) perciving its environment through Sensors.
- ii) Acting upon that environment through actuators.

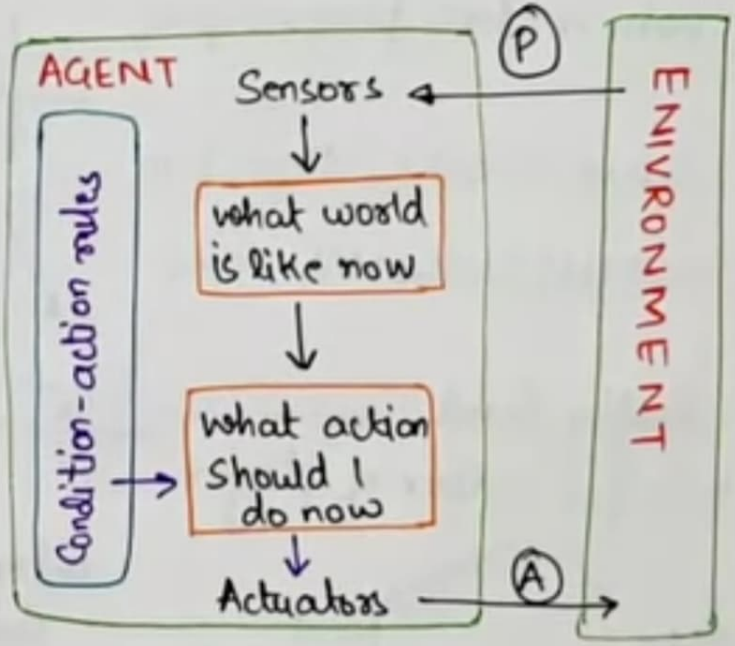
Agent → Info^n gathering.
→ action perform.



① Simple Reflex Agents:

Works only on Current situation/ perception and ignores the history of previous State.

↳ Condition-Action Rule — True



Limitations:-

↳ i) Very limited Intelligence.

ii) No knowledge about non-perceptual parts of State

iii) Can go into infinite loop.

(2) Model based Reflex Agent:

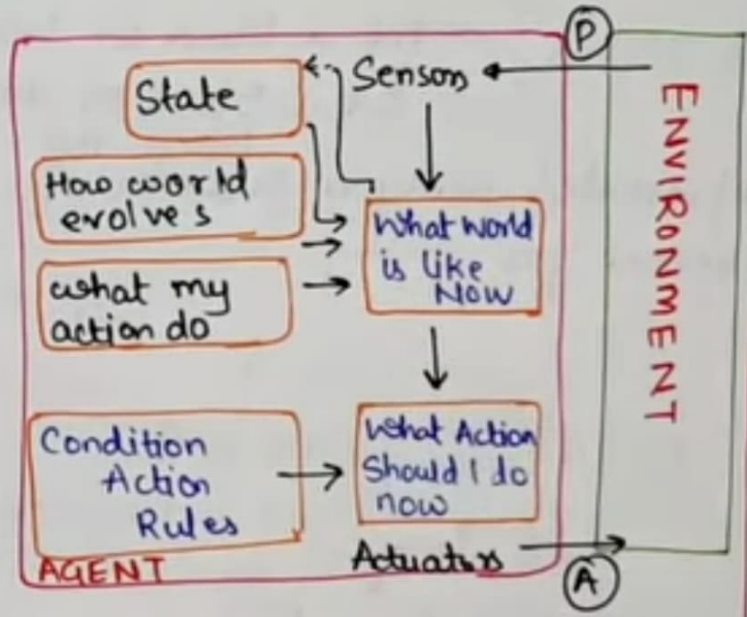
↳ Works by finding the rule whose cond^n matches Current Situation.

↳ Can work in partially observable env., and track situation.

↳ Agent keeps track of internal state which is adjusted by each percept and that depends on percept history.

↳ Model: How things happen in world.

↳ Agent State updation required Info^n:
- (1) How world is evolving
- (2) How agent's action affect the word.



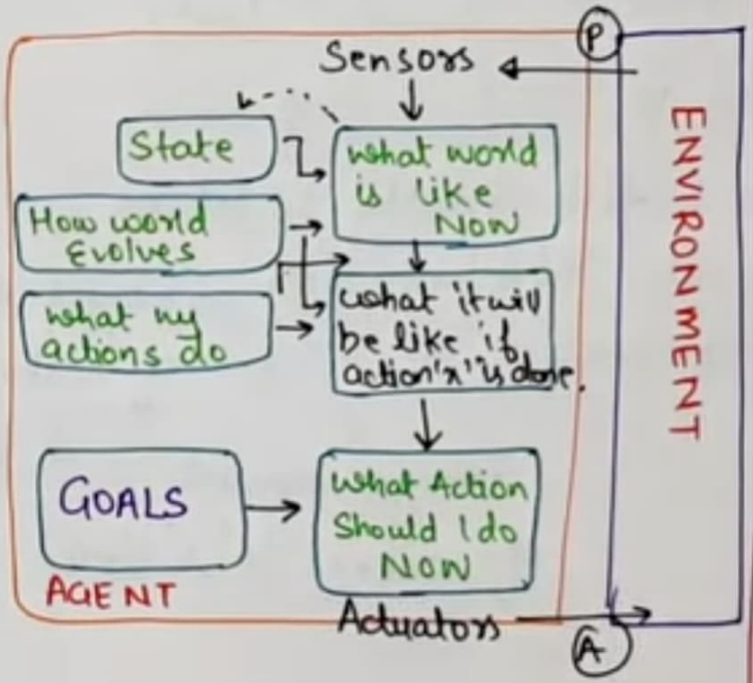
③ Goal-Based Agents:

↳ Focuses only on reaching the goal set.

↳ Agent takes decision based on how far it is currently from the Goal State.

↳ Every action is taken to minimize distance to Goal State.

↳ More flexible Agent.

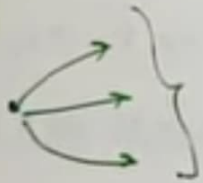


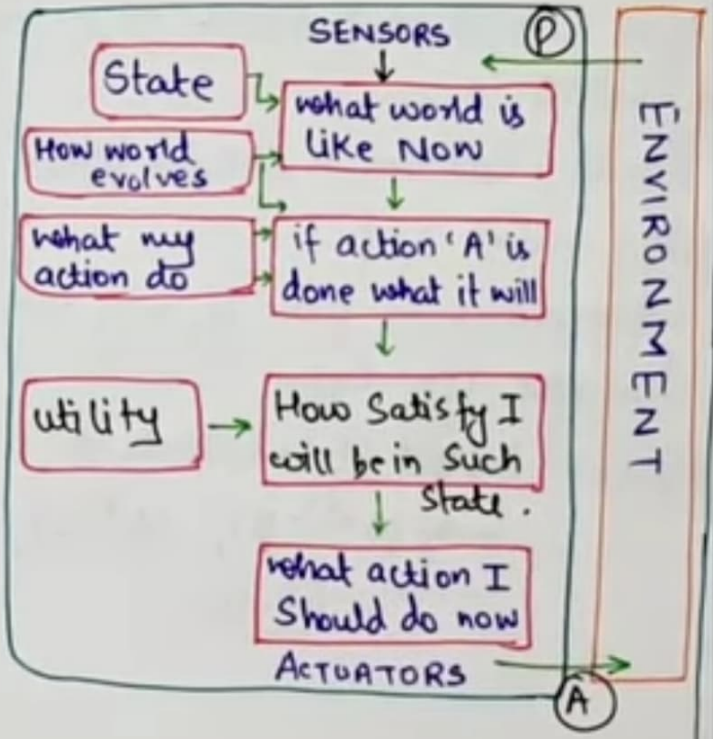
(4) Utility-Based Agents:-

↳ Agents are more Concerned about the utility (Preference) for each state.

↳ Act based not only on goals but also the best way to achieve goal.

↳ Useful when there are multiple possible alternatives and agent has to choose in order to perform best action.

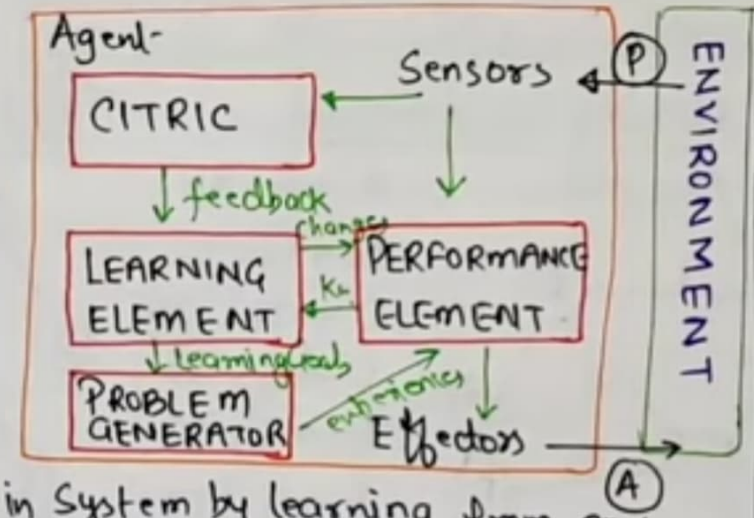




Example of Uninformed Search Algorithms

(5) Learning Agents:-

↳ Can Learn from its past experiences.
↳ Starts to ACT with basic knowledge and then able to act by adapting Learning.



Components:-

(i) Learning Element: Makes Improvement in System by learning from env.

(ii) Critic: Gives feedback about Agent's Performance based on Standard.

(iii) Performance Element: Selects the action to Perform

(iv) Problem Generator: Suggests the action.] new info^n gain.

PEAS: Grouping of AI Agents:

↳ used to group Similar type of agents together.

↳ PEAS:-

- **Sensor:-** Devices from which agent perceives observation from env
- **Actuators:-** devices, H/w, S/w through which Agent performs action on env.
- **Performance measure**
  - It is the o/p we get from an Agent.
  - ↳ Results obtained after Agent Processing.
- **Environment**
  - ↳ All Surrounding things and conditions.

Example:- SELF DRIVING CAR

(P):- Confort, Safety, Time, Legal driving.

(E):- Condⁿ of Roads, Crossings, Traffic Signals

(A):- Steering, Brakes, Horn, Accelerator.

(S):- Camera, GPS, Speedometer, odometer etc.

Classification of Environment in AI:

Environment is part of the Universe that surrounds Intelligent System. → whole part of

① **Accessible and Inaccessible:** env. may not be
↳ Agent can obtain complete and in reach time accurate info about state's env.
↳ Room with temp. as state (accessible)
↳ Event on Earth [Inaccessible]. → nature of env. can't be decided by agent alone. [Stochastic].

② **Deterministic or Non-Deterministic**
↳ Agent's current state and selected action can completely determine env. next state. ↳ doesn't have to worry about uncertainity.

③ **Static or Dynamic:**
↳ Env. doesn't change its state with passage of time.
↳ State of env. changes w.r.t time.

Static:- crossword Puzzles.
Dynamic: Car driving.
↳ Random in nature.

(4) Discrete or Continuous:- observations are not Continuous.

- Finite no. of percepts and actions → CHESS GAME.
- Agent can perceive and make observations from the env. continuously without any lag [Self-driving]

(5) Observable or Partially observable: ⟶ In this some part of env. is not in the reach of agent.

- Agent Sensor can access complete state of env. at each point of time. (fully or completely)

Ques:- Explain Formal, Mundane and Expert tasks in AI.

Humans Learns Mundane (Ordinary) tasks
Since their birth
→ Learn by: Perception, Speaking, using languages, locomotives
→ Easiest to lean.

Formal and Expert task are learn later in the order.

Mundane:- ① Perception (Computer Vision, Speech, Voice.)
② NLP (understanding Lang., generate Lang., Translation.)
③ Reasoning ④ Robotics (locomotive)

Formal Task:- (Verification, Theorem Proving.)
→ Mathematics
→ Geometry.
→ logic
→ Game-theory (Go, Chess, Checkers)

Expert Task:- Engineering.
→ Manufacturing
→ Monitoring
→ Scientific, financial, Medical.

Ques:- What do you mean by uncertainity? why uncertainity arises?

Uncertainity is defined as the Lack of exact info^n or knowledge that helps us to find correct conclusion. } definition.

Uncertainity may be caused by problems with data such as:-

- ① Missing / unavailable data
- ② Unreliable / ambiguous data
- ③ Imprecise / Inconsistent rep^n of data
- ④ Guess based
- ⑤ Default based

] data.

Causes of uncertainity:-

① Uncertain Inputs
- Missing Data
- Noisy data

② Uncertain Knowledge
- multiple causes leads to multiple effects.
- Incomplete knowledge of Casuality in domain.

③ Uncertain outputs
- Abduction, induction are uncertain
- Default reasoning
- Incomplete deduction, inference.

Ques:- what do you mean by Turing Test? How it can be performed?

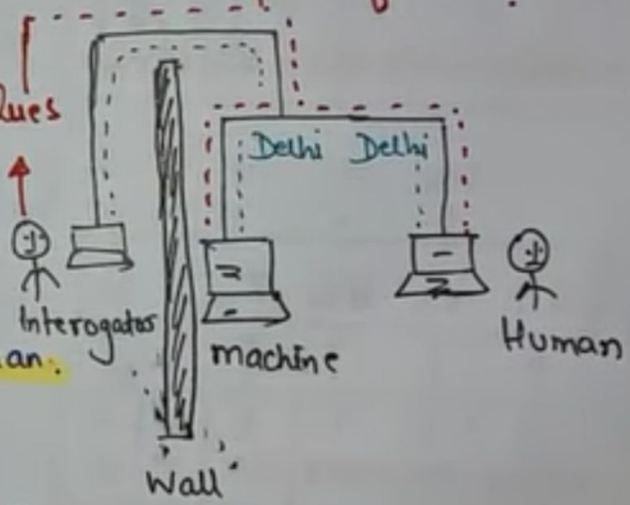
Coined by Alan Turing in 1950.

"Turing test is used to determine "whether or not machines can think intelligently like humans".

Basic Configuration:-

↳ There will be a human Interrogator on one side of wall and other side a machine and human.

↳ Machine Intelligent ⇒ when human interrogator can't distinguish response given by machine and human.



→ Examiner

↳ Knowledge Base.

] machine has passed the test and it is intelligent.

Ques:- What do you mean by Chinese Room Test? Explain how it can be performed?

↳ Also known as Chinese Room Argument.

↳ Proposed by Mr. John Searle in 1980.

↳ Argued that "Turing Test could not be used to determine "whether or not machine is considered as Intelligent".

According to John Searle a machine could pass Turing Test simply by manipulating symbols, without any understanding of those symbols.

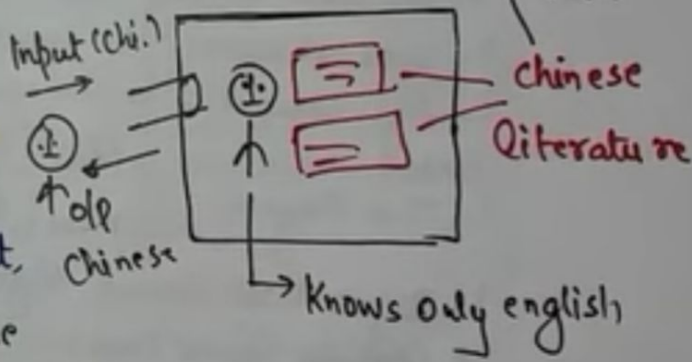
↳ A person/machine can be considered as Intelligent, if and only if they have understanding of what they are doing.

Capital of India? → Delhi ← database

BASIC CONFIGURATION:-

↳ A person knowing english not chinese sits in room with huge volume of Chinese literature.

↳ Chinese symbol ϕ, return φ.
↳ " " ϕφ, return æ. } Rules



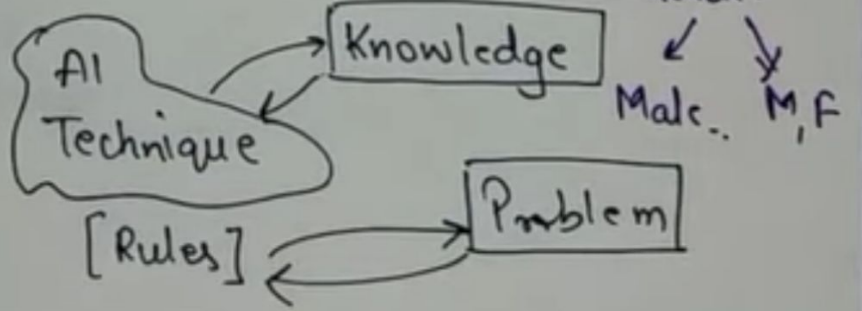
AI Technique: It is a method that exploits knowledge that should be represented in such a way that:

- i) Knowledge Captures Generalization
- ii) Understandable by People.
- iii) Easily modifiable to Correct
- iv) Can be used in many situations
- v) Can reduce its volume.

Parts of AI Technique:-

1. **Knowledge Representation:** Used to Capture the knowledge about Real world.
2. **Search Algorithm:** Finding / Searching Solution of the Problem.

Knowledge
- Large in Volume
- Not well formatted
- Constantly changing



BFS:- Breadth First Search.

↳ Explores all the nodes at given depth before proceeding to the next level.

↳ Uses Queue to implement.

ALGORITHM:

i) Enter Starting nodes on Queue.

ii) If Queue is empty, then return fail and Stop.

iii) If first element on Queue is GOAL Node, then return Success and Stop.

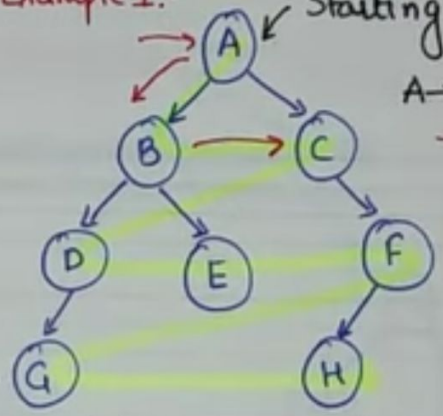
ELSE

(IMP)

iv) Remove and expand first element from Queue and place children at end of Queue.

v) Goto Step (ii)

Example 1:-



Starting Node.

A → B → C → D → E → F → G → H

Initial Queue {A} = Goal node ✗

↳ Remove from Queue and add its successor to Queue.

{B, C}

↓ → (D, E)

{C, D, E}

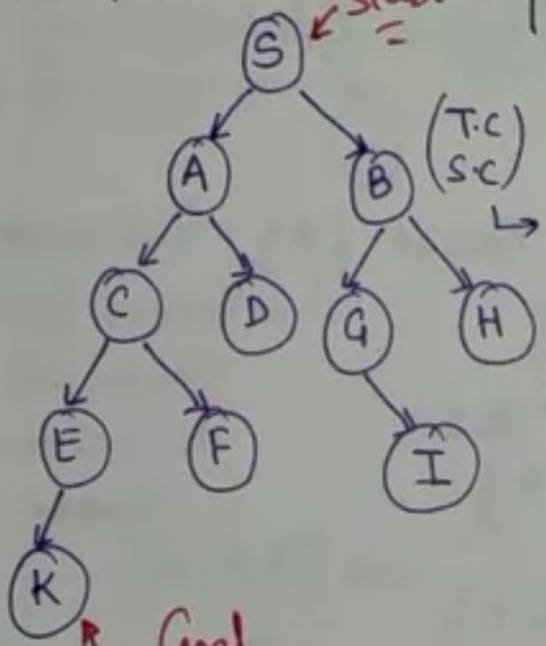
↓

{D, E, F} → {E, F, G}

↓

{G, H}

Example 2:-



(T.C
S.C
↳ $O(b^d)$
↳ depth
↳ branching factor

{S}
↳ {A, B}
↳ {B, C, D}
↳ {C, D, G, H}
↳ {D, G, H, E, F}
↳ {H, E, F, I}
↳ {F, I, K}

S → A → B → C
↓
D
↓
G → H → E → F → I → K ↳

Adv:- find a Soln if its exists.
↳ minimal Soln in Least no. of steps.

Disadv:- More memory
↳ need lots of time
↳ Soln is far from root node.

DFS: Depth-first Search.

↳ Recursive algorithm.

↳ Starts from root node and follows each path to its greatest depth node before moving to the next path.

↳ Implemented using STACK. (LIFO)

ALGORITHM:

i) Enter Root node on Stack (Push)

ii) Do until Stack is not empty

↳ a) Remove Node (Pop)

↳ i) If Node = Goal Stop.

ii) Push all children of node in Stack

Adv:- Less memory

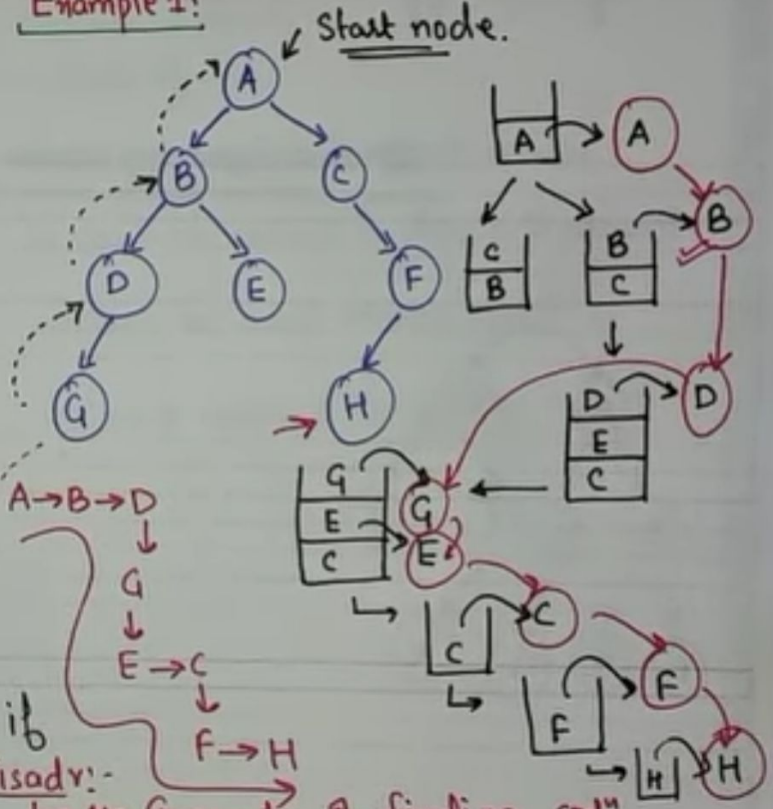
↳ Less time to reach goal node if traversal in right path.

Disadv:-

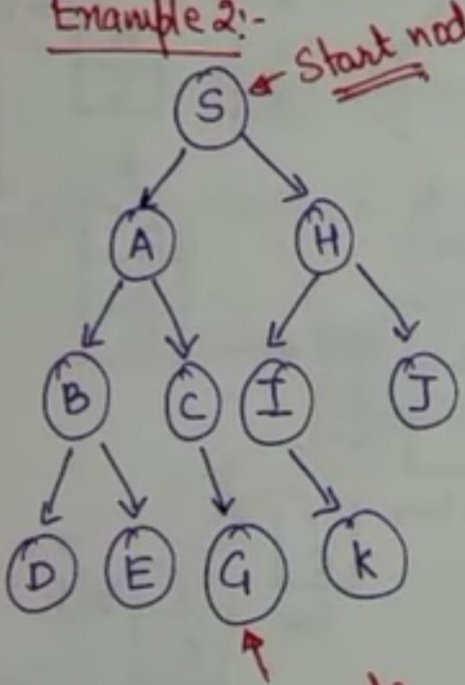
↳ No Guarantee of finding sol".

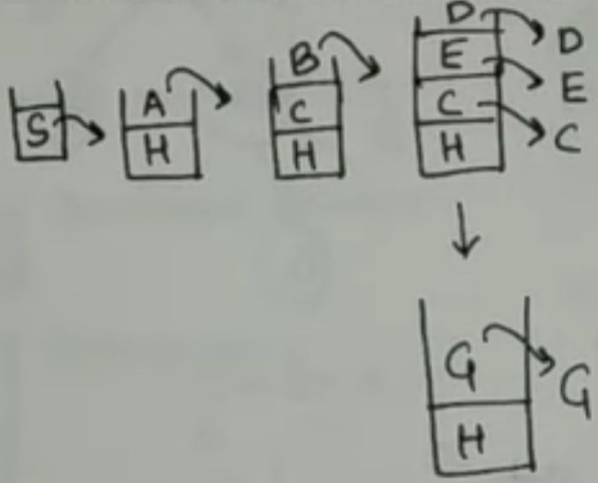
↳ Infinte loop.

Example 1:



Example 2:-





S → A → B → D → E → C → G

Time Complexity = $O(b^d)$ ↳ depth.
↳ branching factor

Space Complexity
↳ $O(bd)$

Depth-Limited Search Algorithm:

↳ Working is Similar to DFS but with a predetermined limit.

↳ Helps in Solving the problem of DFS: Infinite Path

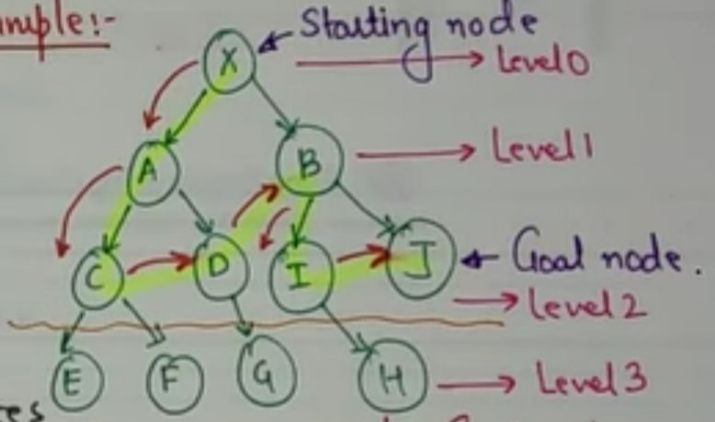
Termination Conditions:
- (i) Failure Value: There is no solution.
- (ii) Cutoff Failure: Terminates on reaching predetermined depth. } no Soln. (IMP)

Advantages:
↳ Memory Efficient.

Disadvantages:-
↳ Can be terminated without finding Soln. → Incompleteness
↳ Not optimal.

[Time Complexity: $O(b^d)$
[Space Complexity: $O(b \times d)$

Example:-



[Depth, d = 2]

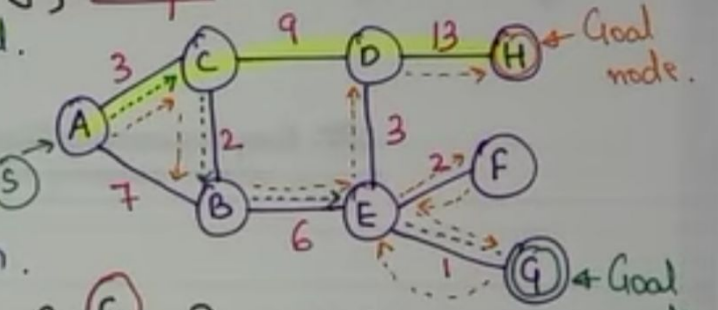
X → A → C → D → B → I → J

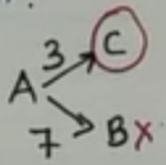
Path.

This depth will result in no Soln.

Uniform Cost Search Algorithm: {Backtracking} **Example:-**

- Used for weighted Tree/Graph Traversal.
- GOAL is to path finding to goal-node with lowest Cumulative cost } optimal Path.
- Node expansion is based on path costs.
- Priority Queue is used for implementation.
  - High Priority to minimum Cost.





Path to Goal node (G)

$A \xrightarrow{3} C \xrightarrow{2} B \xrightarrow{6} E \xrightarrow{1} G$

(12)

Goal node 'H'

$A \xrightarrow{3} C \xrightarrow{2} B \xrightarrow{6} E \xrightarrow{1} G \xrightarrow{1} E \xrightarrow{2} F \xrightarrow{2} E \xrightarrow{3} D \xrightarrow{13} H$

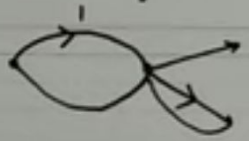
(33)

Advantage:

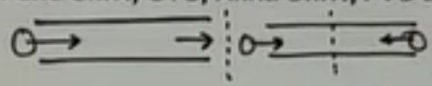
- Optimal Soln.

Disadvantage:

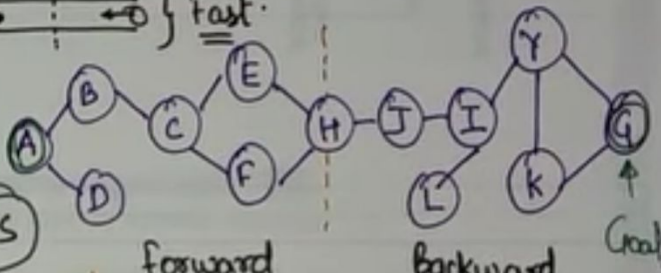
- Struck in Infinite Loop.



Bidirectional Search Algorithm:

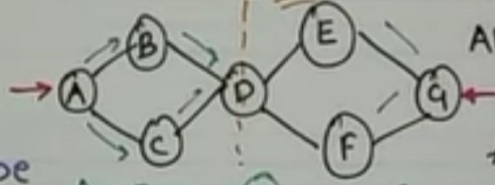


↳ Two different Searches are run Simultaneously.
- → Forward Search (Start → Goal)
- → Back Search (Goal → start)



↳ Single Search Graph is replaced with two small graphs.

↳ Any Search technique can be used. (BFS, DFS . . . .)



Forward	Backward
A	G
ABD	GKY
ABDC	GKYI
ABDCEF	GKYILJ

A → B → C → D

G → E → F → D

↳ Search STOP Condition: when Graphs intersect

ABDCEFH GKYILJH

A → B → C → D → F → E → G

A → B → D → C → E → F → H → J → L → I → Y → K → G

Advantages:- i) Fast
iii) Less Memory

Disadvantages:- i) Implementation is difficult
iii) Goal State ~~shou~~ should be Known in advance

Iterative Deepening Depth-First Search:

↳ Combination of both DFS and BFS.

↳ Best Depth Limit is found out by gradually increasing limit. Initially d=0 (every iteration increase by 1.)

Advantage: {Fast, Less Memory}

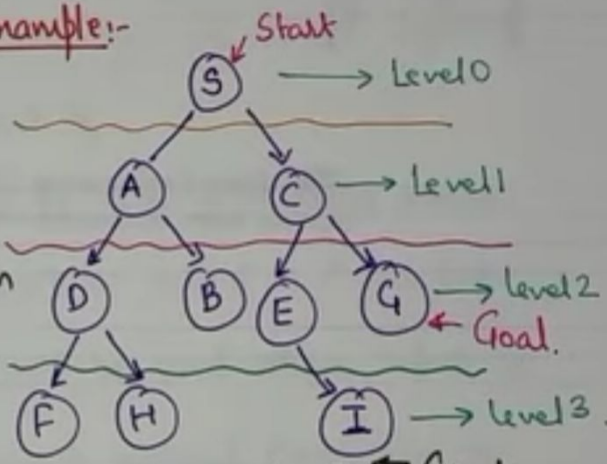
↳ Incorporates benefits of both BFS and DFS.

Disadvantage:

↳ Repeat the work/process.

STACK

Example:-



1st Iteration, d=0 [S]

2nd Iteration, d=0+1=1 [S → A → C]

3rd Iteration, d=1+1=2 [S → A → D → B → C → E ↓ G]

4th Iteration, d=2+1=3 [_____] G]

Informed Search: → Heuristic Search, Heuristic function.

↳ Information about GOAL State is present.

↳ Better-than UnInformed Search.

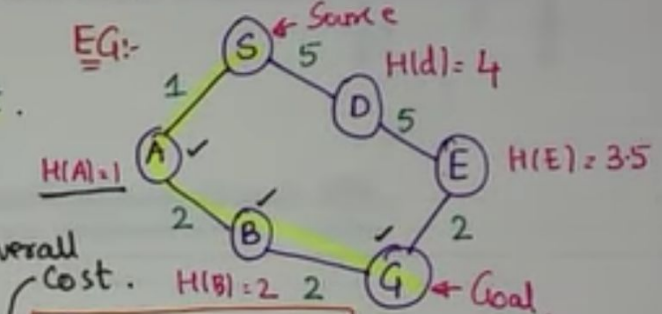
↳ finds optimal Sol^{n} to reach GOAL State.

↳ Using **HEURISTIC FUNCTION**

min. Path cost.

↳ It is a Search which tries to reduce amount of Search that must be done by making intelligent choices for the nodes that are selected for expansion.

↳ Eg:- i) Best Fit Search ALGO
ii) A* Search ALGO.



overall Cost.

$f(n) = G(n) + H(n)$ → for every node.

$G(n)$ → Path Cost

$H(n)$ → Heuristic Value

(A) $f(n)$ (1) + (1) = (2)

S → (A) $f(n) = 1+1 = 2$ (1)
S → D $f(n) = 5+4 = 9$ (5)

2 < 9

A → B $[f(n) = 3+2 = 5]$

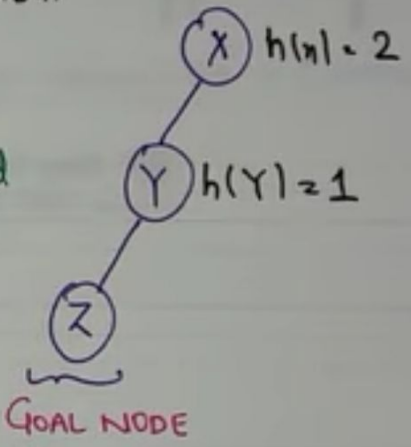
S →(1) A →(2) B
(5) ↓2
G

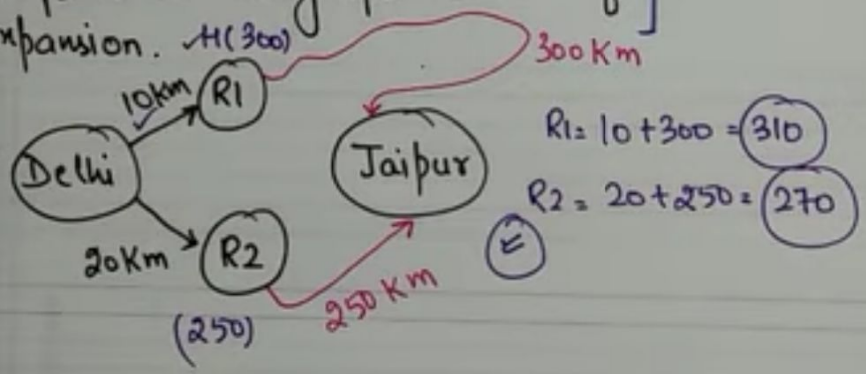
Heuristic Search:- → Tries to Solve Problem in minimum steps/cost.

↳ Tries to optimize a problem using heuristic function. [Informed Search].

Heuristic Function: It is a function H(n), that gives an estimation on the cost of getting from node 'n' to the GOAL State. } estimated Value.

[Helps in Selecting optimal node for expansion.





Types of Heuristic:- (underestimate)

↳ (i) **Admissible:** In this Heuristic function, never overestimates the cost of reaching the Goal. $H(n) \leq H'(n)$ {Goal}

↳ h(n) is always less than or equal to actual cost of lowest-cost path from node 'n' to goal.

(ii) **Non-Admissible** :- Overestimate $H(n) > H'(n)$

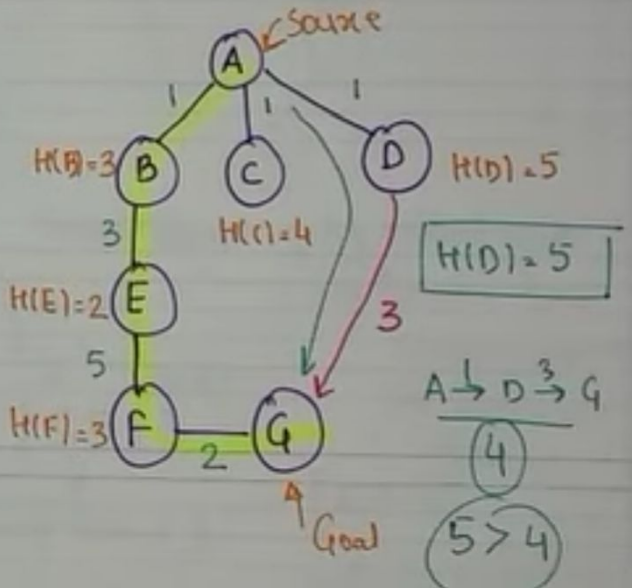
$H(B)=3$, $H(C)=4$, $H(D)=5$

$f(n) = H(n) + G(n)$

B = 4, C = 5, D = 6.

Actual = (11)

3 < 11



Ques:- Differentiate b/w blind and heuristic Search.

Blind Search:- It is also known as unKnown/uninformed Search.
- ↳ There is no info^n about the Searching.
- ↳ No Knowledge of where the GOAL.
- ↳ Eg:- Depth first, Breadth first Search
- ↳ Efficiency is low
- ↳ Slower than Heuristic
- ↳ Large memory is used
- ↳ No func^n (special) is used.

Heuristic Search:- It is a method of solving problems more easily and fast. They have Knowledge of where goal or finish of the graph. (Informed Search)

Eg:- Hill climbing, A*, AO*
- ↳ Highly efficient < less time / Less cost
- ↳ finds sol^n Quickly.
- ↳ no large memory is required.
- ↳ Heuristic func^n is used.

BEST FIRST SEARCH: GREEDY SEARCH. [BFS / DFS]

↳ Uses evaluation Algorithm (func^n) to decide which adjacent node is most promising and then explore.

↳ Category of Heuristic or Informed Search.

↳ Priority Queue is used to store Cost of nodes. → Sorted order.

ALGORITHM:

Priority Queue 'PQ' Containing initial States

Loop

if PQ = Empty Return FAIL

Else

NODE ← Remove_First(PQ)

if NODE = GOAL

Return Path from Initial to NODE

Else

Generate all Successors of NODE and insert newly generated NODE into PQ according to COST Value

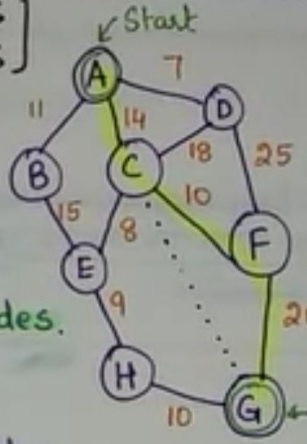
END LOOP

Space Com:- $O(b^d)$

T.C:- $O(b^d)$

b: branching factor

d: depth



Straight Line distance.

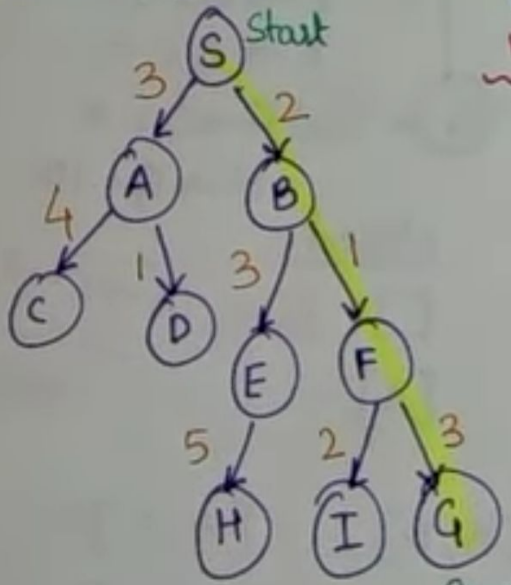
A → G = 40
B → G = 32
C → G = 25
D → G = 35
E → G = 19
F → G = 17
G → G = 0
H → G = 10

open	closed
[A]	[]
[C, B, D]	[A]
[B, D]	[A, C]
[F, E, B, D]	[A, C]
[E, B, D]	[A, C, F, G]

Path = A → C → F → G

(44)

Best First Search Example:



node(n)	H(n)
A	12
B	4
C	7
D	3
E	8
F	2
H	4
I	9
S	13
G	0

Open	Closed
[B, A]	[S]
[F, E, A]	[S, B]
[G, I, E, A]	[S, B, F]
	[S, B, F, G]

Path = S → B → F → G

BEAM SEARCH:

optimized Version of **Best First Search** { Only Predetermined no. of Best Partial Solutions are kept as Candidates. }

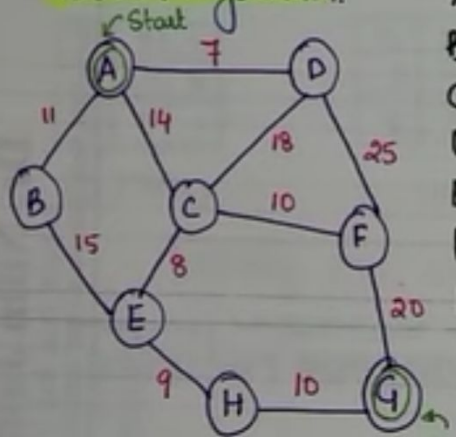
↳ Heuristic Search Algorithm.

↳ Explores a Graph by expanding the most promising node in a "limited set".

↳ Reduces Memory Requirement.

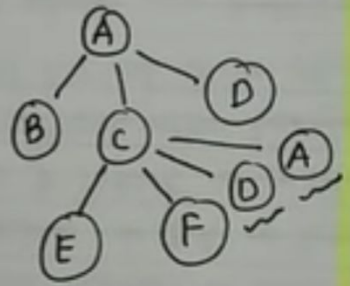
↳ GREEDY Algorithm.

Beam Value (β) = ? → Predetermined no. of best partial soln are kept as Candidates.

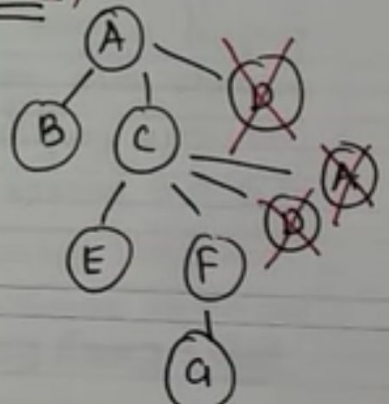


A → G = 40
B → G = 32
C → G = 25
D → G = 35
E → G = 19
F → G = 17
H → G = 10
G → G = 0

Best-First Search.



Beam-Search (β = 2)



Ques:- Explain A* Algorithm for Search.

↳ Uses heuristic func^n $h(n)$ and cost to reach the node 'n' from start state $g(n)$.

↳ Finds shortest path through search space.

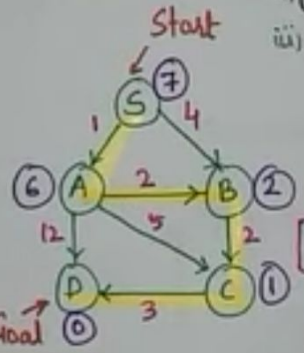
↳ Fast and optimal result.

$f(n) = g(n) + h(n)$ ⟶ heuristic Value (Child node)

$f(n)$: estimated cost. $g(n)$: ↳ Cost to reach node

ALGORITHM:

↳ i) Enter Starting node in OPEN list.
ii) If OPEN List is Empty return FAIL
iii) Select node from OPEN list which has smallest Value (g+h).
↳ if node = Goal, return Success
iv) Expand node 'n' and generate all Successors
↳ Compute (g+h) for each Successor node.
v) if node 'n' is already in OPEN/CLOSED, attach to back pointer.
vi) Go to iii)



S→A = 1+6 = 7 ✓
S→B = 4+2 = 6 ✓
S→B→C = 4+2+1 = 7 ✓
S→B→C→D = 4+2+3+0 = 9 r1
S→A→B = 1+2+2 = 5 ✓
S→A→C = 1+5+1 = 7 ✓
S→A→D = 1+12 = 13 r2
S→A→B→C = 1+2+2+1 = 6 ✓
(#) S→A→B→C→D = 1+2+2+3 = 8 r3
S→A→C→D = 1+5+3 = 9 r4

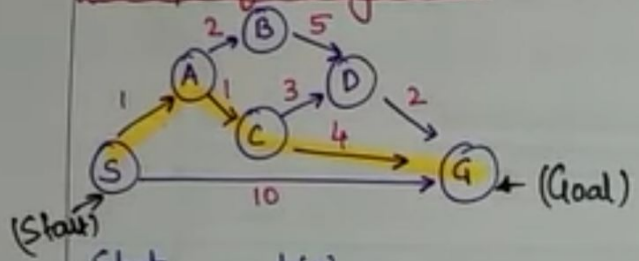
Advantages:-

↳ i) Best Searching algorithms.
ii) optimal and complete.
iii) Solving Complex problems.

Disadvantages:-

↳ i) Doesn't always produces shortest.
ii) Complexity issues.
iii) Requires memory.

Example of A* Algorithm:-



State	h(n)
S	5
A	3
B	4
C	2
D	6
G	0

$S \to A = 1+3 = 4$

$S \to G = 10+0 = 10$

$S \to A \to B = 1+2+4 = 7$ ✓

$S \to A \to C = 1+1+2 = 4$ ✓

$S \to A \to C \to D = 1+1+3+6 = 11$ ✓

$=$ $S \to A \to C \to G = 1+1+4 = 6$.

$S \to A \to B \to D = 1+2+5+6 = 14$ $=$

$S \to A \to C \to D \to G = 1+1+3+2 = 7$

$S \to A \to B \to D \to G = 1+2+5+2 = 10$

Ques:- Describe how Problem Reduction is done with AO* Algorithm.

AND-OR Graphs:- useful for representing the problem solution that can be solved by decomposing them into smaller set of problems, all of which must be solved.

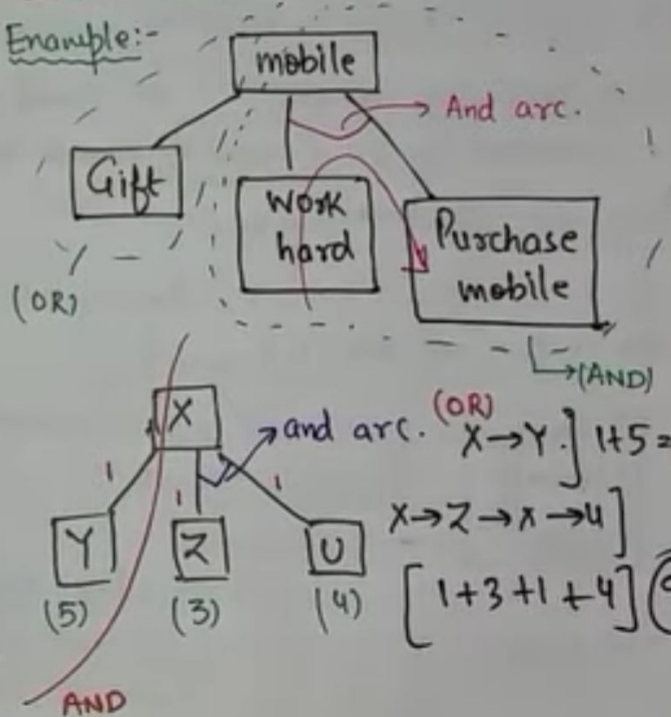
→ ANDARCS. } May point to 'n' no. of successor nodes.

↳ Also like Best-Fit Search can be used that has the ability to handle AND Arcs.

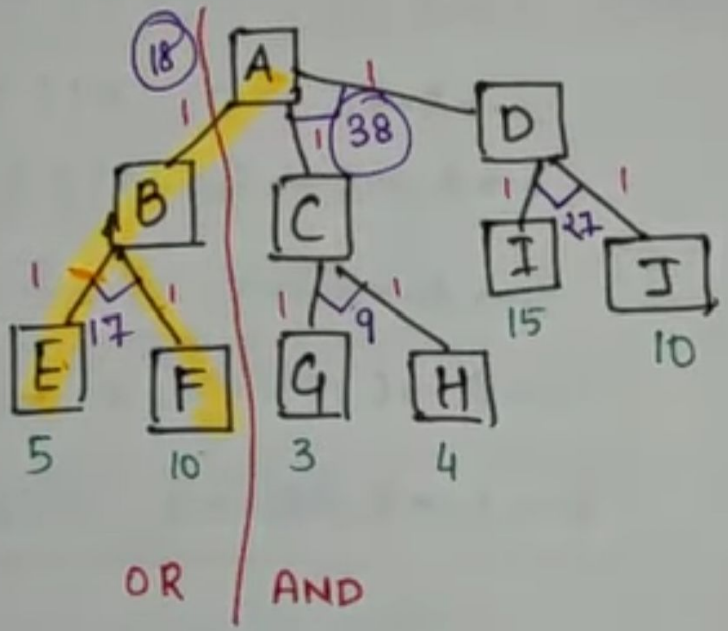
[FUTILITY: Should be chosen to correspond to a threshold value. if Est. cost > futility. Stop Search.]

OR

Example:-



Example AO* Algo:-



Ques:- Write Short note on "Hill-Climbing Search"

↳ Variant of generate and test method in which feedback from test procedure is used to help generator decide which dir^n to move in Search Space.] always moves in a Single dir^n.

↳ It is like DFS }

↳ Local Search algo]
↳ Greedy Approach] No Backtracking.

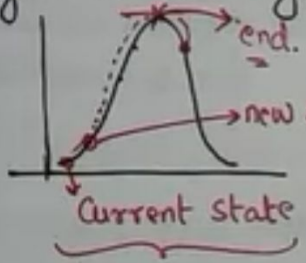
Eg:-

1	2	4
5		7
3	6	8

(Starting State)

(4) (5) (X) (6)

↳ we will choose this

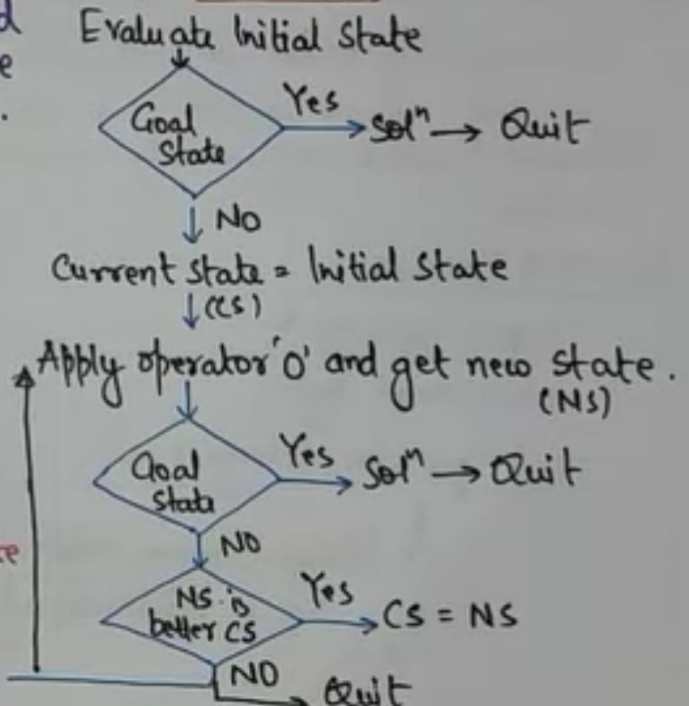


if new state is better than Current state.

↓

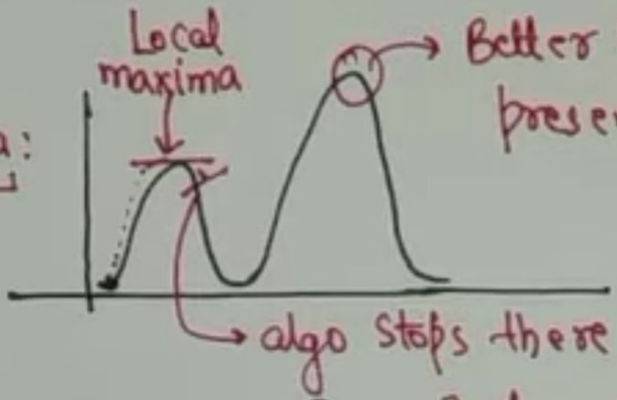
new state = Current state

Flowchart :-

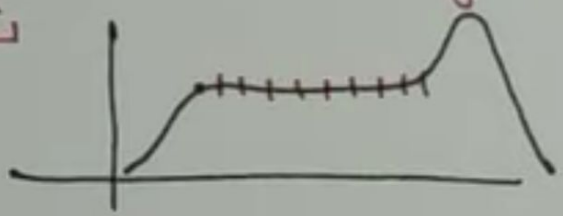


Limitations:-

① Local maxima:

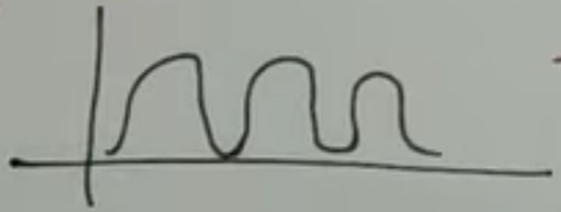


② Plateau:



algo gets Same Value/ Soln for some time.

③ Ridge:-



all Successor gets Same Value.

Example:-

Ques:- Describe how the branch and bound technique could be used to find shortest Sol^n. What are the disadvantages of it? [Only Fwd Dir^n] ↳ Parallelization is difficult.

Branch and Bound Search is a way to Combine Space Saving of depth-first search with heuristic info^n. ↳ limited to small size N/W.

↳ optimal Sol^n is chosen among many Sol^n paths.

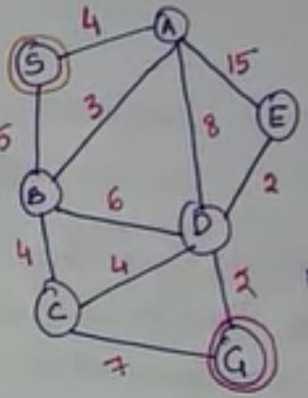
↳ Idea in Branch-and-Bound Search is to maintain lowest-Cost path to a GOAL found so far, and its Cost.

↳ Typically used with depth-first Search.

↳ Two Parts → Branch:- Several choices are found.

→ Bound:- Setting Bound on Sol^n Quality.

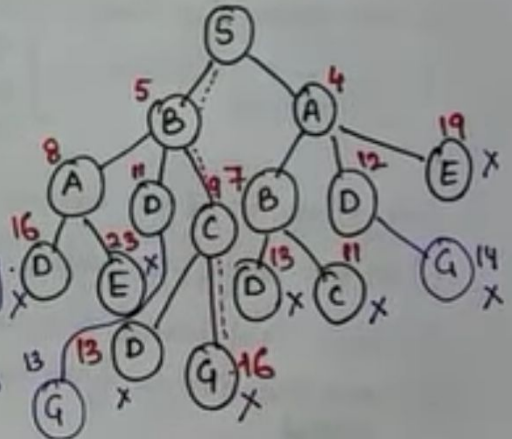
↳ Pruning:- Trimming off branches where Sol^n Quality is Poor.



Bound1 = 16

Bound2 = 13

Optimal Path
↳ S → B → D → G
13



Simulated Annealing:- Checks all neighbors.

↳ Simulated Annealing (SA) allows downward steps.

↳ Annealing is a process in metallurgy where metals are slowly cooled to make them reach a state of low energy where they are very strong.

Advantages	Disadvantages.
→ Easy to code for complex problems also → Gives Good Soln. → Statistically guarantees finding optimal soln.	→ Slow Process. → Can't tell whether an optimal soln is found. ↳ Some other method is also req.

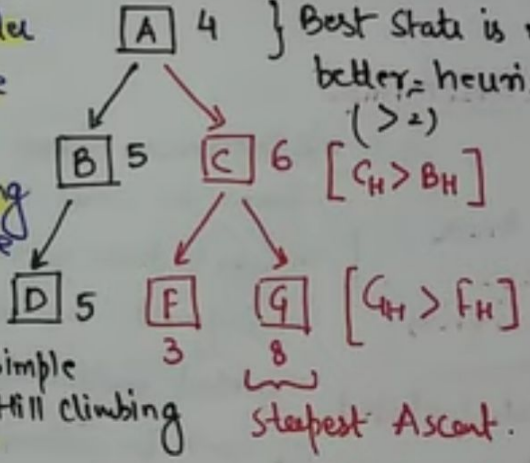
Simulated Annealing	Hill Climbing
i) annealing schedule is maintained.	X
ii) Moves to worst states may be accep.	X
iii) Best state found so far is also maintained	X

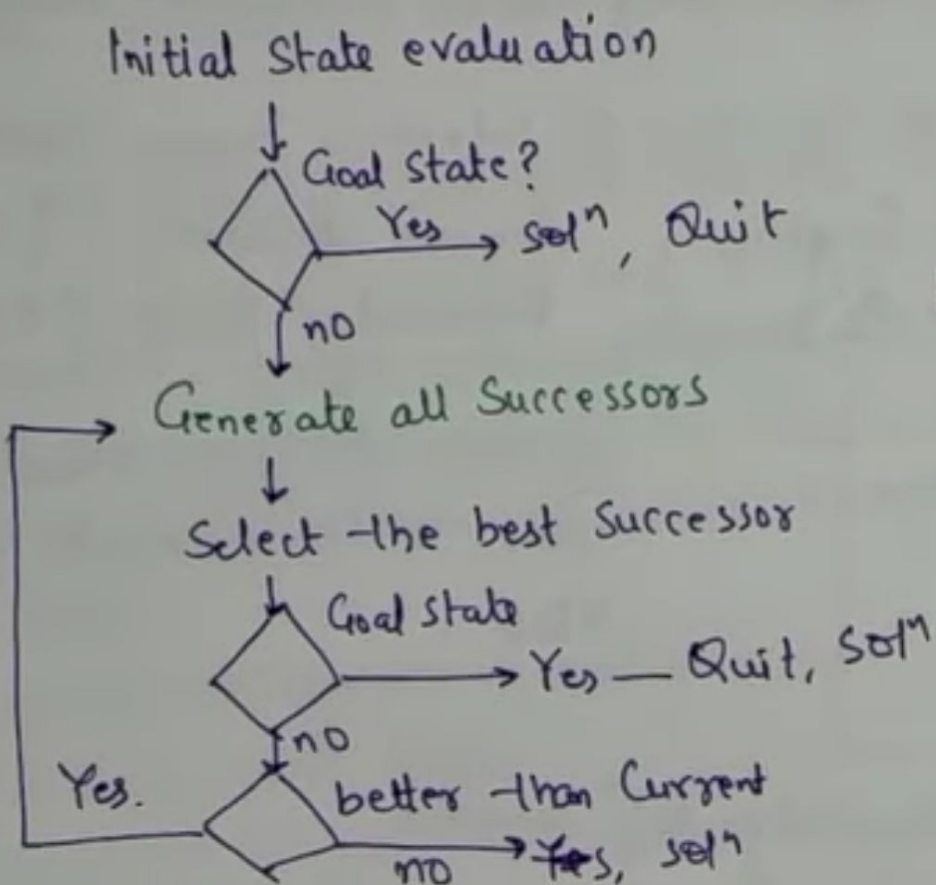
Ques:- Differentiate Simple Hill climbing and Steepest-Ascent Hill climbing. Write down the Algorithm for Simple Hill Climbing. → Video.

In Steepest-Ascent Hill climbing multiple points are checked.

In Simple Hill Climbing first State which is better than Current State is selected and rest of the States are not explored whereas in Steepest Ascent Hill Climbing Algo. selects the best among the children states that are better than the Current State.

- ↳ All moves are Considered and best one is selected as next State.
- ↳ examines all neighboring nodes and selects nodes closest to Soln as next node.





Flowchart.

State Space Search: Used in Problem Solving.

↳ It is a process used in A.I in which Successive Configurations or States of an instance are considered with intention of finding a GOAL State with desired property.

EIGHT TILE PUZZLE

Start(S)

1	4	3
2		5
8	6	7

GOAL(G)

1	2	3
8		4
7	6	5

↳ Problems are modelled as **State Space**

↳ Representation:

S: (S, A, Action(S), Result(S,a), Cost(s,a))

- S: set of all possible states.
- A: Set of all possible actions
- Action(S): Func" [which action is possible for Current state]
- Result(S,a): Func" [State reached by performing action 'a' on state 's)]
- Cost(s,a): Costing.

State Space: set of states in which a problem can be.

Actions Possible: 'Right'

- up
- down
- Left
- Right

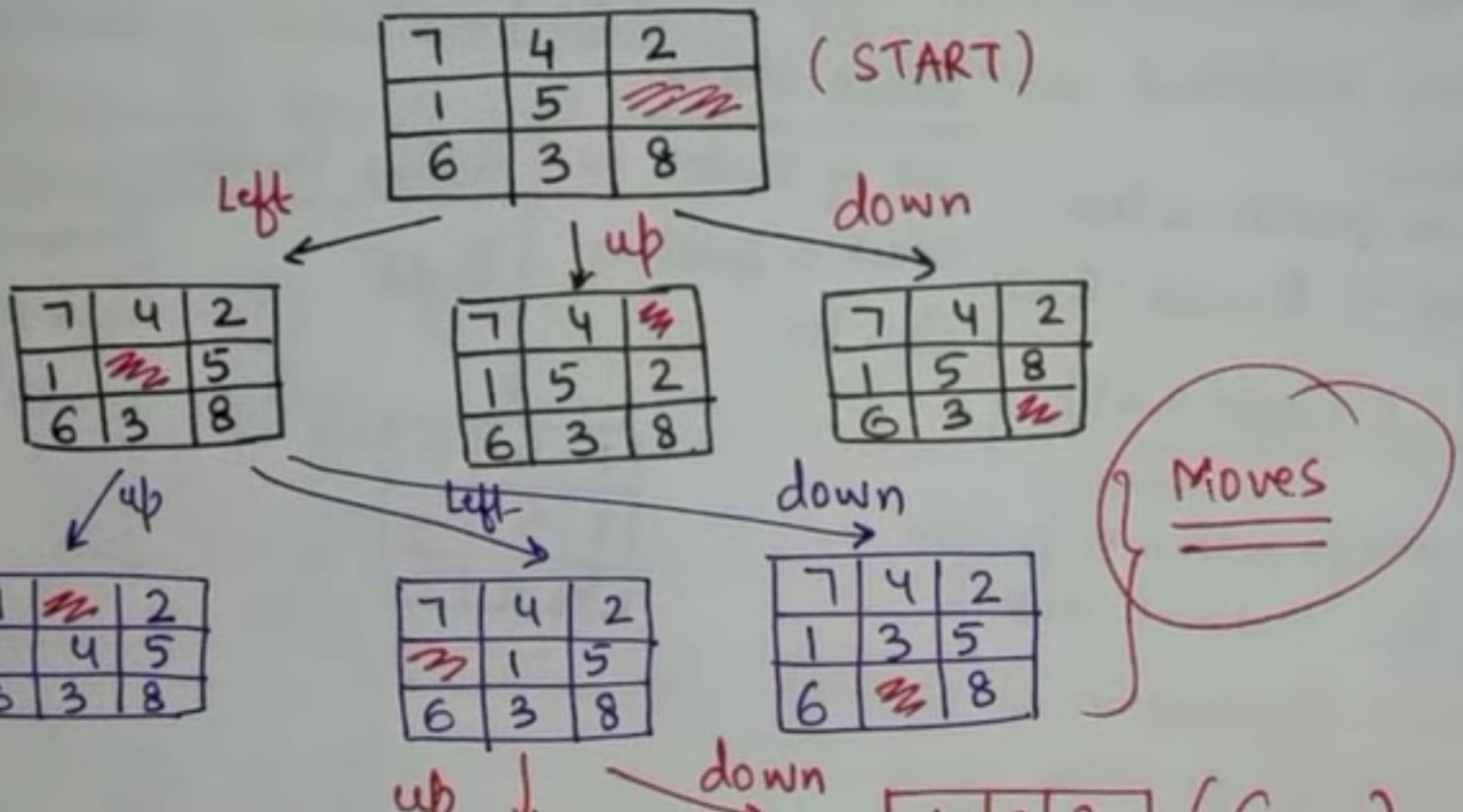
Legal moves for a state.

1	4	3
2	5	
8	6	7

right is not possible now

- up
- down
- left

new state(S1)


(START)
Left
up
down
Moves

Propositional Calculus:-

↳ It is a system that deals with the method used for manipulation of the symbols according to some rules.

X: It is Hot
Y: It is Humid
Z: It is raining

{ All, Some → (PL) x }

ALPHABET SET:

↳ i) Set of Variables or Propositional Symbols P, Q, R
ii) Logical Constants → True (T), false (F)
iii) Two Parentheses "(" and ")" (X, Y)
iv) Set of logical operators

IMP:- ↳

	C word	symbol.	example
i)	not	¬	¬X
ii)	and	∧	X∧Y
iii)	or	∨	X∨Y
iv)	implies	→	(X→Y) if X then Y
v)	if and only if	↔	(X↔Y)

(1) if it is humid ('Y') then it is hot. (X)
↳ (Y → X)

(2) if it is hot (X) and humid (Y) then it is not raining (Z).
(X∧Y) → ¬Z

vi) Set of equivalence relations or laws: (P, Q, R) are Variables.

- Commutative Laws: $P \wedge Q \cong Q \wedge P$, $P \vee Q \cong Q \vee P$
- Associative Laws: $(P \wedge Q) \wedge R \cong P \wedge (Q \wedge R)$, $(P \vee Q) \vee R \cong P \vee (Q \vee R)$
- Double Negation: $\sim(\sim P) \cong P$
- De-Morgan's Law: $\neg(P \vee Q) = \neg P \wedge \neg Q$, $\neg(P \wedge Q) \cong \neg P \vee \neg Q$ (Imp)
- Absorption Law: $P \wedge (P \vee Q) \cong P$, $P \vee (P \wedge Q) \cong P$
- Law of Contradiction: $P \wedge \neg P \equiv$ False $\begin{matrix} P=1 \\ \neg P=0 \end{matrix}$ ⓪ $\begin{matrix} P=0 \\ \neg P=1 \end{matrix}$ ⓪
- Law of excluded middle: $P \vee \neg P =$ True $\begin{matrix} P=1 \\ \neg P=0 \end{matrix}$ ① $\begin{matrix} P=0 \\ \neg P=1 \end{matrix}$ ①
- Law of impotency: $P \wedge P \cong P$ $\left[\begin{matrix} P=1 \to 1 \\ P=0 \to 0 \end{matrix}\right]$

RULES OF INFERENCE:

→ ① MODUS PONENS: If 'P' and 'P → Q' is given to be true, then we can infer that 'Q' is true.

P: It is a holiday ✓ (T)
Q: The school is closed] → we can infer that it is true.
P → Q: If it is a holiday, then School is closed ✓ (T)

② MODUS TOLLENS: If '~Q' and '~P → ~Q' are given to be true, then we can infer that ~P is true.

~Q = School is not closed.
if it is not a holiday, then School is not closed.
(~P → ~Q)
→ ~P] it is not a holiday (True).

Ques:- Define Tautology and truth table.

Truth Table shows how the truth or falsity of a Compound Statement depends on the truth or falsity of Simple Statements.

A **Tautology** is a formula which is always true. ↳ opposite is Contradiction (always false)

Eg:- Show that $(P \to Q) \vee (Q \to P)$ is tautology.

P	Q	$P \to Q$	$Q \to P$	$(P \to Q) \vee (Q \to P)$
T	T	T	T	T
T	F	F	T	T
F	T	T	F	T
F	F	T	T	T

all are true

Some of Truth Table Eg:-

① Negation:-

P	~P
T	F
F	T

Simple Statements.

② AND: $(P \wedge Q) \to$ True when P and Q both are true.

P	Q	$P \wedge Q$
T	T	T
T	F	F
F	T	F
F	F	F

→ Compound Statement

Representing Simple Facts in FOPL: → models world in terms of objects.

Real-world facts can be represented as logical propositions written as well-formed formulas in propositional logic.

Symbols:-
1. Objects
2. Properties
3. Relation.

} Representation.

Symbols are formed of the following:-

1. Set of all uppercase english Alphabets.
2. Set of digits from 0 to 9.
3. Underscore character.

[All dogs are brown] Can't be written in propositional logic
↳ first-order predicate logic.

Defining a Sentence :-

Every **atomic sentence** is a Sentence.
↳ is defined as predicate constant of arity 'n', followed by $t_1, t_2, \ldots t_n$ terms enclosed in parentheses and separated by Commas.

1. if 'S' is Sentence then $\neg S$ is sentence.
2. if S1, S2 are sentences] $S1 \wedge S2$ (Conjuction) ✓
3. " "] $S1 \vee S2$ (disjunction) ✓
4. " "] $S1 \rightarrow S2$ (implication) ✓
5. " "] $S1 \equiv S2$ (equivalence) ✓
6. if x is Var. and s is Sentence the $\forall x$ s is a Sentence.
7. " " $\rightarrow \exists x$ s is a Sentence.

Quantifiers in Predicate Calculus:-

There are two Quantifiers used in first-order predicate calculus:-

1. Universal] for all 'x' such that — $\forall x$ ($\rightarrow$)
2. Existensial] for some 'x' such that — $\exists x$ ($\wedge$)

↳ Constrain the meaning of sentence containing a Variable.

↳ Quantifier is followed by a Variable and a Sentence.

① All Boys like football. } → $\forall x$: Boys(x) → like(x, football)

② Some Boys like football. } → $\exists x$: Boys(x) ∧ like(x, football)