INTRODUCTION (Chp 1)

AI Dimensions

- → Acting Humanly – Turing Test Approach
- → Thinking Humanly – cognitive Modeling Approach
- → Thinking Rationally – 'laws of Thoughtful' Approach
- → Acting Rationally – Rational Agent Approach.

Modeling (user work)
- Thought processing / reasoning
- Behavior / Action

Evaluation (client work)
- Performance measure (accuracy, success rate)
- Success acc to human standards vs. success acc. to ideal concept of intelligence.

- Building intelligent entities
- To augment / complement human Intelligence

→ Purpose of AI
- To augment human-thinking
- use comp's AI to understand how human's think.

WHAT IS AI?

Views of AI falls in 4 categories:

① Thinking humanly — cannot be a definition of AI b/c we'll need to see how every brain works & then we need to evaluate & model.

without phy interaction (turing test) we can not deduce a machine → Need to do pyschological experiments &

② Acting humanly — Query de ke ans lelena is not humanly. Need to percieve obj too.

③ Thinking rationally — The way we used to solve prob using discrete notations. (formal statement)
Real life data set is never complete!!
↓ incomplete data set se kabhi bhi rational statement create nai kr sakte for all possible queries & never 100% accurate.

④ Acting rationally — knowing what it knows & working right acc to it — right definition.
(doing right thing given what it knows)
↓ working right acc. to the knowledge provided to the

Major component of AI (basic)

Knowledge → backend pe knowledge provide krein so that it works accordingly.

Reasoning → on what basis the machine is working (backend reasoning — why is knowledge provided

Day/ Date:

Language understanding — Subdomain ~~Nation~~ Natural lang. processing (NLP)
→ Able communicate & know diff. lang.

Learning — Subdomain of AI → when machine comes in new environment, it is able to take ~~knowledge~~ decisions acc to knowledge provided acc. to new patterns.
→ Able detect new patterns.

Rational Agent
↳ as per given environment gives best performance.
Agent → an entity that acts. It percieves the environment & then performs actions.
Ideal performance || expected performance
↓ 100% accurate
↓ what we've taught acc. to that what accuracy we expect from our class.

1.1.1 - Turing Test Approach
Designed to provide a satisfactory operational definition of intelligence.
A comp passes test if a human interrogator, after posing some written questions, cannot tell whether the written responses come from a person or a computer.
The computer would need to have following capabilities:
→ Natural Language Processing ~~to~~ enable it to communicate successfully in Eng.
→ Knowledge Representation to store what it knows or hears.
→ Automated Reasoning to use the stored info to ans ques. and to ~~know~~ draw new conclusions
→ Machine learning to adapt to new circumstances & to detect & extrapolate patterns.

→ Turing Test avoids physical interaction but includes videosignal for the perceptual abilities.
The comp also needs: Computer Vision — to percieve objects
Robotics — to manipulate obj & move about.

Dazzle

(Rough) ①

Day/ Date:

Task Environment categorization

{ how many times environment is being changed/updated?

→ Episodic vs. Sequential

↳ one action is not depending on another (environment) not agent

Seqn.

one move is dependant on another → 'Chess' → eg.

Static vs. Dynamic

↳ environment is not changing ~ constant → ~~checks~~ ~~crossboard~~ crossword puzzle

↳ environment keeps on changing

semi-dynamic — In chess environment same, score change

Discrete vs. Continuous

↳ finite range of no. " no. of step

→ cannot tell distinct → continuous range of no of step

↳ don't know after how many steps we'll get the result.

Known vs Unknown

↳ we know the rules of an environment (if prior knowledge is given)

→ rules are not known before. Machine will have to learn itself

TABLE DRIVEN AGENT

Most impracticed approach → table lookup for entire history

machine } agent
AI app }

FOUR TYPES IN ORDER OF INCREASING GENERALITY:

① Simple Reflex Agents

(If-else) → select action on basis of current percept, ignoring all past percepts

fixed no. of action & environment state → aur uski basis pe If-else implement

Dazzle

Day/ Date:

→(percept history ke mutabiq kaam karna rahe)

② Model Based Reflex Agent → (percept history ko dekh ke action)

Maintains internal state that keeps track of aspects of env. that cannot be currently observed

Standby pe rakhein ke jab unchein ho too clean kre warna standby

③ Goal based agent

we should know goal criteria → what is the destination

→ The agent uses goal info to select b/w possible actions in the current state.

④ Utility Based → Efficient

The agent uses a utility func. to evaluate the desirability of states that could result from each possible action.

→ at cost we need to reach our goal; accuracy/efficiency/max. performance goal

Learning Agents

4 things that make a perfect agent

① Critic

② Learning Element

③ Prob generator → taking relevant action to explore & learn (to make machine learn) (kis tarah machine ko learn krwana hai)

④ Performance agent ↳ sensor se percieve krein envir. & give action acc. to the history

→ providing feedback acc. to your current action how is it working (machine) how good/bad agent is performing & how it can be ~~evaluated~~ improvised.

→ to take feedback & improvise the agent acc. to it to better performance.

Day/ Date: Chp 3

→ Define a Problem is necessary before making state space (explore every option while making a state space)

√ state space → (eg. map)

① initial state + ② action + ③ where is state taking (transition)

Eg. left, right, clean → vaccum cleaner → 2 boxes □ A □ B

→ define goal, initial & states → 8 possible

Eg 2 sliding puzzle

→ can move left, right, up, down

Initial

2	8	3
1	6	4
7	–	5

Goal

1	2	3
8	–	4
7	6	5

↓

①(a)

2	8	3
1	6	4
–	7	5

①(b)

2	8	3
1	6	4
7	5	–

①(c) 2

N Queen prob

n×n → grid ; if n=4 ; place 4 queens in 4×4 grid in such a way that they don't attack each other.

		Q_3	
Q_1			
			Q_4
	Q_2		

	Q_2		
			Q_4
Q_1			
		Q_3	

→ 2 possible ways

→ we'll do by backtracking

Day/ Date:

INTELLIGENT AGENTS (Chp 2)

2.1 – AGENTS & ENVIRONMENTS

An agent is anything that can be viewed as percieving its environment through sensors & acting about that environment through actuators.

Percept is used to refer to agent's perceptual input at any given instant.
An agents ~~perceptual~~ percept sequence is the complete history of everything the agent has ever percieved.
(an agent's choice of action at any given instant can depend on the entire percept but not on anything it hasn't percieved yet)

- An agent's behaviour is described by the agent function that maps any given percept sequence to ~~any~~ an action.
- The table is an external characterization of the agent
- Internally, the agent function for an artificial ~~intelligence~~ agent will be implemented by agent prog.

2.2 – GOOD BEHAVIOR – THE CONCEPT OF RATIONALITY

Rational Agent is the one that does the right thing — (as per given environment, performs best)
When an agent is plunked down in an environment, it generates a sequence of actions acc. to the percepts it recieves & these actions causes the environment to go through seq. of states.
If desirable performance → performed well.
→ This notion of desirability is captured by a performance measure that evaluates any given sequence of environment states.

2.2.1 – Rationality

What is rational at any given time depends on 4 things:
① → Performance measure that defines the criterion of success.

②, The agent's prior knowledge of environment.
③, The actions that agent can perform.
④, The agent's percept sequence to date.

Rational Agent:

* for each possible percept seq, a rational agent should select an action that is expected to max. its performance measure, given the evidence provided by the percept seq & whatever built in knowledge the agent has"

Priori – The 'geography' of environment.

2.2.2 – Omniscience, Learning & Autonomy

Omniscient Agent – knows the actual outcome of its actions & can ~~relate~~ act accordingly; but omniscience is impossible in reality.

- Rationality is not same as perfection → perfection maximizes actual performance.
  → rationality " expected "

Info gathering – Doing actions in order to modify future percepts. Exploration is also important.
↳ should be done in an initially unknown environment.

→ Agent's initial config reflect prior knowledge, but as it gains (learns) experience this may be modified & augmented.
→ A rational agent should be autonomous – it should learn what it can to compensate for partial or incorrect prior knowledge.
→ Task Environment – The probs to which the rational agents are the solutions.
↳ various flavours: The flavour of task environment directly affects the appropriate design for agent program.

Day/ Date: ____________

*(to write case study acc. to PEAS & describe task environment)

2.3.1 – Specifying The Task Environment

PEAS (Performance, Environment, Actuators, Sensors)

Agent Type – Medical Diagnosis system

P – Healthy ~~well~~ patient, reduced cost

Envir. – Patient, hosp, staff

A – Display of qs, test, diagnosis, treatment, referrals

S – keyboard entry of symptoms, finding, patient's ans.

2.3.2 – Properties of Task Environments

Rationality – Machine is rational if it is capable to gather info, store & predict/detect the pattern

Task environment categorization – Category of environment & reasoning.

Environment Specify – To plan the environment

These dimensions determine the appropriate agent design & the applicability of each of the principal~~s~~ families of techniques for agent implementation.

• Fully Observable vs. Partially Observable

Fully:

• If an agent's sensor give it access to the complete state of environment at each point in time then it is fully observable.

→ If sensors detect all relevant to the choice of action. (depends on performance measure)

→ convenient b/c the agent does not need to maintain any internal state to keep track of the world.

Partial:

B/c of noisy, inaccurate sensor or b/c parts of state are simply missing from sensor data.

eg. ~~total~~ vaccum cleaner with a local dirt sensor cannot tell if there is dirt in other sq.

Unobservable:

If agent has no sensor.

Day/ Date:

- Single Agent vs. MultiAgent

- Single-Agent: An agent solving a crossword puzzle itself.
- MultiAgent: An agent playing chess in a two agent environment.

① * Describe how an entity should be viewed as an agent.

→ chess is a competitive multiagent; b/c one tries to max it's performance & min other's.

→ Taxi driving is partially cooperative & partially competitive; max performance of all agents.

② * Agent-design prob: (i) communication often emerges as rational behaviour in multiagents.

(ii) In some competitive environments, randomized behavior's rational b/c avoids pitfalls of predictability

- Deterministic vs. Stochastic

If next state of environment is completely determined by the current state & action executed by the agents, then we say the environment is deterministic, otherwise stochastic.

→ taxi driving stochastic b/c cannot predict traffic

Uncertain ~ If not fully observable or not deterministic

Stochastic implies → uncertainity about outcomes is quantified in terms of probabilities.

non-deterministic ~ Actions are characterized by their possible outcomes, but no probabilities are attached to them.

- Episodic vs. Sequential

Episodic — ~~An~~ agent's environment divided into atomic ep. Agent recieves a percept for each ep & then performs one action/ep & not dependant on actions taken in previous episodes.

eg. spot defective parts — Simpler Environment

Sequential — One move dependant on another; short-term actions can have long term consequen

eg. chess & taxing driving — Chess

→ Episodic simpler b/c doesn't need to think of future.

Day/ Date:

- Static vs. Dynamic

Dynamic – If environment can change while an agent is deliberating eg. Taxi driving.

Static – Environment is not changing – constant. – easy & donot need to worry about time. eg. crossword

Semi-dynamic – If environment doesn't change, but agent's performance score is changing. eg. chess.

- Discrete vs. Continuous

Applies to the state of environment, to the way time is handled & to the percepts & actions of agent.

eg. chess has finite no. of discrete states, set of percepts & actions.

– Taxi driving has a continuous speed & location.

- Known vs. Unknown

– Doesn't refer to environment but to agent's state of knowledge about 'laws of phy' of environment.

Known – The outcomes (or probabilities) for all actions are given.

- It is possible for a known environment to be partially observable. eg. ~~sola~~ solitare card games.

Unknown – the agent will have to learn how it works in order to make good decisions.

- Unknown can be ~~partia~~ fully observable eg. video game (screen shows the entire state but dk what the buttons do)

– Environments are drawn from environment class & environment generator which selects particular envir.

2.4 – THE STRUCTURE OF AGENTS

Design an agent prog. that implements the agent function – mapping from percepts to actions.

agent = architecture + program.

2.4.1 – Agent Programs

Take current percept as ~~internet~~ input from sensors & return an action to actuators. (b/c nothing more available in environment)

Agent function ⟶ takes entire percept history.

Day/ Date:

4 basic types of agents: (see diag of these from book/slides)

① Simple-reflex agent ② Model based reflex agent ③ Goal based agent ④ Utility based agent

Learning agents – That can improve performance so as to generate better actions.

2.4.2 – Simple Reflex Agents

Simplest – selects actions on the basis of current percept, ignoring the rest of the percept history. (if-else condition)

→ fixed no. of actions & environment state (ussi ki basis pe if else implement)

eg. vaccum cleaner ; complex → Automated Taxi driving

↓

condition-action rule: when an unexpected action triggers some established

(if car-in-front-is-braking then initiate-brake) ← connection in agent prog.

Escape from infinite loops is possible if agent can randomize its actions.

2.4.3 – Model-Based Reflex Agents

Effective way to handle partial observability is for agent to "keep track of the part of the world it can't see now". ie. Agent should maintain some sort of internal state that depends on the percept history & thereby reflects atleast some of the unobserved aspects of the current state.

> updating this internal state info with time requires:

① Need info about how world evolves independently of the agent.

② Need info about how the agent's own actions affect the world.

→ Model: 'knowledge about how world works' ; An agent that uses such agents is called 'model based agent'

2.4.4 – Goal Based Agents

The agent also needs some sort of goal info that describes the situations that are desirable.

Dazzle

Day/ Date:

→ Action selection is straightforward — when goal is satisfied by a single action.

→ Action selection can be tricky — so we do search & planning to achieve goal.

→ Appear less efficient, but it's flexible b/c knowledge that supports decision is explicit & can be modified.

& ~~can~~ action can be easily changed.

2.4.5 — Utility Based Agents (utility — quality of being useful) — efficient approach

Utility Function — An internalization of performance measure.

→ The agent has utility function to evaluate the desirability of states that could result from each possible ~~state~~ action.

→ Has advantages in terms of flexibility & learning.

① goals are inadequate, but a utility based agent can still make rational decisions.

② when many goals to reach, utility provides a way in which the likelihood of success can be weighed against the importance of goal.

* at cost we need to reach our goal; efficiency / accuracy / max performance goal.

→ A rational utility based agent chooses the action that maximizes the expected utility of the action outcomes

i.e. the utility that agent expects to derive given the probabilities & utilities of each outcome.

2.4.6 — Learning Agents

→ Takes in accounts the amount of work & concludes a desirable method.

Advantage: Allows agent to operate initially in an unknown environment & then become competent.

Four components:

① Learning Element ② Performance Element ③ Problem generator ④ Critic

• Learning element → Responsible to take feedback from critic & improvise agent acc. for better performance

• Performance element → selecting external actions; our entire agent → takes in percept & decides on action.

• Critic → necessary b/c percepts themselves provide no indication of agent's success.

↳ how well agent is doing wrt a fixed performance standard.

• Problem Generator Responsible for suggesting actions that will lead to new & informative experience.

↳ taking relevant actions to explore & learn to make machine learn.

Dazzle

Day/Date:

SOLVING PROBLEMS BY SEARCHING (Chp 3)

- In which we see how an agent can find a sequence of actions that achieves its goals when no single action will do.

→ Goal based agent called prob solving agent. - use atomic representation; states of world are considered as whole with no internal structure visible to the prob solving agents.

3.1 — PROBLEM-SOLVING AGENTS

→ Intelligent agents are supposed to maximize their performance measure.

Goal formulation → based on current situation & the agent's performance measure, 1st step of prob solving

Prob formulation → process of deciding what actions & states to consider; given a goal.

* search — process of looking for a sequence of actions to reach the goal.
* search algo takes prob as input & return a solution in form of action sequence
* after sol. found, the action it recommends can be carried out; this is called execution phase.

(formulate → search → execute)

While agent is executing the sol seq, it ignores the percepts when choosing action b/c it knows in advance what it will be — this is called 'open-loop system'.

3.1.1 — Well defined Problems & Solutions

Defining a prob is neccessary before making state space; explore every option while making a state space.

state space = initial state + action + where is state taking (transition) } eg. map

> Problem can be solved by 5 components:

① Initial state → The state at which agent starts from.

② A description of possible actions.

③ A description of what each action does; formal name for this is transition model.

Dazzle

Day/ Date:

→ State space forms a directed network or graph in which nodes are states & link b/w them are actions. A path ~~is in~~ The state space is a sequence of states connected by actions.

④ Goal test — determines whether a given state is a goal state.

⑤ Path cost function — assigns a numeric cost to each path.

The prob solving agent chooses a cost ~~from~~ func. that reflects its own measure performance.

• Step cost of taking action a in state s to reach state s' is → $c(s, a, s')$.

→ sol to prob is an action sequence that leads from initial state → goal state.

→ sol quality is measured by path cost func & optimal sol has lowest cost path.

3.1.2 — Formulating Problems

Abstraction — The process of removing detail from a representation.

↓ we should abstract the state description, as well as the actions.

3.2 — EXAMPLE PROBLEMS

Toy Problems — Intended to illustrate/exercise various problems solving methods. It can be given a concise, exact description & hence is used to compare performance of different algo

Real world prob — Whose sol. people are actually concerned about. Donot have single description, given general formulation.

3.2.1 — Toy Problems

① → Vaccum World

• States → Environment with n ~~states~~ locations has $n \cdot 2^n$ states.

vaccum cleaner has $2 \cdot 2^2 = 8$ states.

• Initial state → Any state can be designated. • Actions: Right, Left, suck/clean → $\boxed{A}\ \boxed{B}$ → 2 box

• Transition model → The actions have their ~~eff~~ expected effects, except that moving left in leftmost, moving Right in rightmost & sucking in clean environment, have no effect.

(draw complete state space)

Dazzle

Day/ Date:

• Goal test → Checks whether all states are clean

• Path cost → Each step costs 1, so path cost = no. of steps in path.

② → Sliding Puzzle / 8 - Puzzle.

7	2	4
5		6
8	3	1

Start state

	1	2
3	4	5
6	7	8

Goal State

• States → Specifies the location of each of the eight tiles & the blank in one of the 9 sq.

• Initial states → Any state can be designated as initial. Any given goal can be reached by half of the possible initial states.

• Actions → Left, Right, Up, Down

• Transition model → (draw state space)

• Goal test → Checks whether the current state matches the goal state.

• Path cost → Each step has cost 1, so path cost = no. of steps in path.

③ → 8 Queen Problems

To place 8 queens on a chessboard such that no queen attacks any other.

2 kind of formulations:

Incremental formulation → Involves operators that augment the state description, starting with empty state; ~~for 8~~ each actions adds a queen to the state.

Complete State formulation → starts with all 8 queens on the board & moves them around.

^ in either case, path cost is of no interest as only the final state counts.

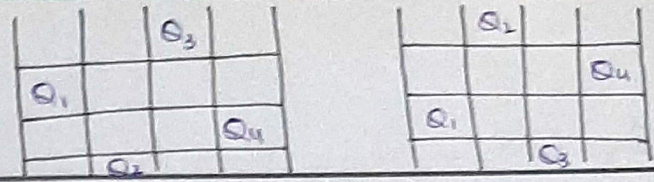
→ In incremental:

• States → Any arrangement of 0→8 queens on board is a state.

• Initial State → No queens on board.

Dazzl

4-Queen prob →



Day/ Date:

2 possible ways ; we'll d by backtrack

- Actions → Add a queen to any empty square.
- Transition model → Returns the board with a queen added to specified sq ; state space.
- Goal test → 8 queens on the board, none attacked.

^ this gives us $64! \approx 1.8 \times 10^{14}$ possible seq to investigate.

3.2.2 – Real world Problems

① Route finding problems → defined in terms of specified locations & transitions along link b/w them.
Route finding algos are used in a variety of applications.

② Touring prob → contains set of cities The agent has visited.

③ Travelling salesperson prob → touring prob that each city must be ~~place~~ visited exactly once.

④ Robot navigation → generalized route finding prob → robot move in continuous space with an infinite set of possible actions & space.

3.3 – ~~[illegible]~~ SEARCHING FOR SOLUTIONS

Uninformed : Blind search – We don't know domain or anything (no knowledge) about problem.
→ we only know the start & goal state – keep exploring to reach goal by comparing at each step

Informed : Heuristic approach – work towards local benefit.
→ To solve prob using knowledge & in polynomial time.

→ Complexity of Uninformed – $O(b^d)$ → b is branching factor., d → depth.
– Non-polynomial → exponential growth. (NP problem)
→ Time consuming → Root first approach. → Optimal sol.

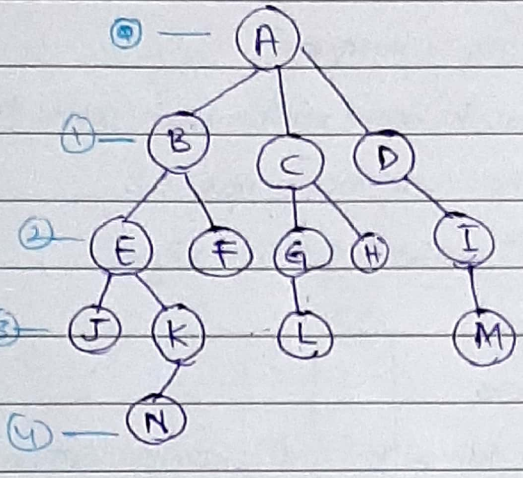
→ Quick but might compromise on optimality of sol.

Day/ Date:

→ Uninformed Search

• Breadth First

* Level by level search
* Complexity → $O(b^d)$ (time + space)
* FIFO
* Shallowest node
* Complete : Yes
* Optimal : Yes

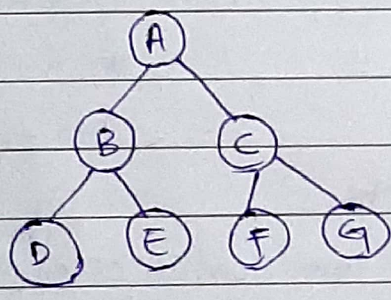


4-levels

~~A~~
~~B~~CD
~~C~~DEF
~~D~~EFGH
~~E~~FGHI
~~F~~GHIJK
~~G~~HIJK
~~H~~IJKL
~~I~~JKL
~~J~~KLM
~~K~~LM
~~L~~MN
~~MN~~
~~N~~

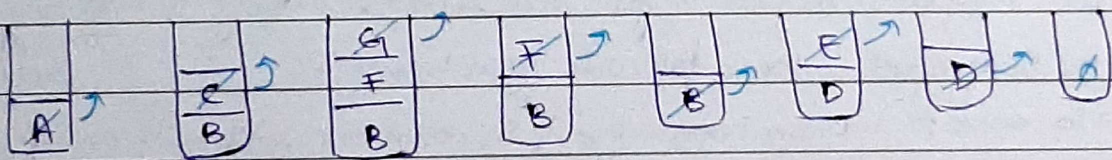
• Depth First

* Depth
* Stack (LIFO)
* Deepest
* Incomplete



complexity: $O(b^m)$

Space: $O(bm)$



A → C → G → F → B → E → D

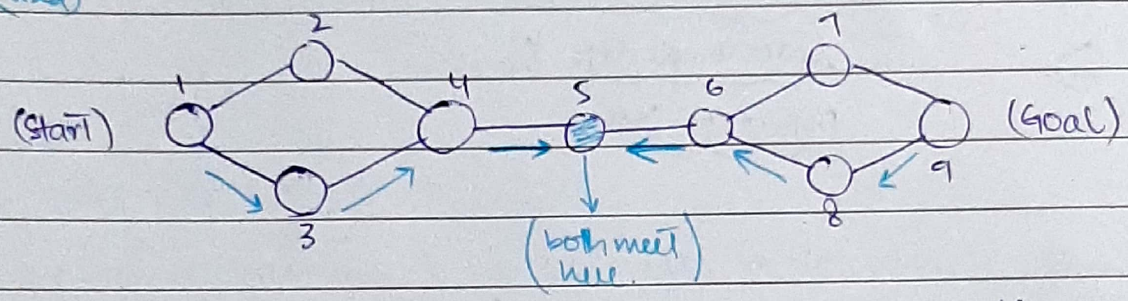
Incomplete means if search space is infinite, it might get stuck in a loop & never give result.

Dazzle

Day/ Date:

→ Bidirectional search

- Simultaneously search from an initial to goal & backward from goal to initial & stopping when two meets.
- Time complexity : $2(b^{d/2})$; Space complexity : $O(b^d)$
- complete in bfs & incomplete in dfs. ; It is optimal.
(used)



→ divides one graph into subgraphs ; can either use bfs or dfs

→ Depth limited search

→ Similar to DFS with a predetermined limit.

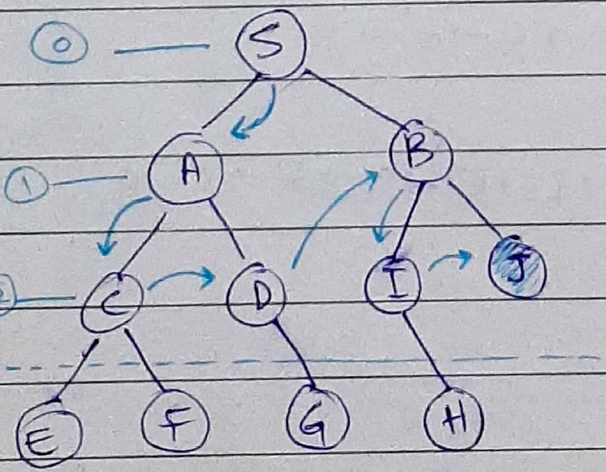
→ can solve the drawback of infinite path in DFS.

→ node with depth limit will act as if it has no successor nodes

→ It can be terminated if

① standard failure value : Indicates that prob doesn't have any sol.

② cutoff failure value : defines no sol for prob within a given depth limit.



Complete : no (if goal beyond l. ($l < d$) or infinite branch

Time = $O(b^l)$

Space = $O(bl)$

Optimal = NO

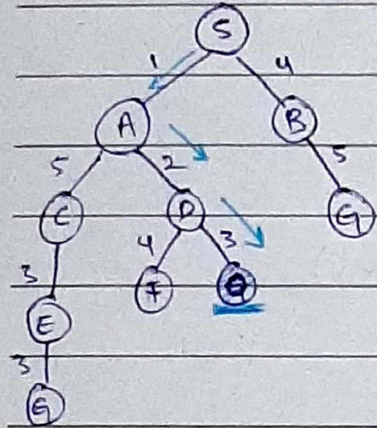
Dazzle

Day/ Date:

→ Uniform-Cost Search

→ find path to goal node which has the ~~First~~ lowest cumulative cost.

→ ~~equivalent~~ equivalent to BFS if path cost of all edges is same.



Completeness: Yes

Time Complexity: $O(b^{[C^*/\epsilon]})$

Space Complexity: (same)

Optimality: Yes

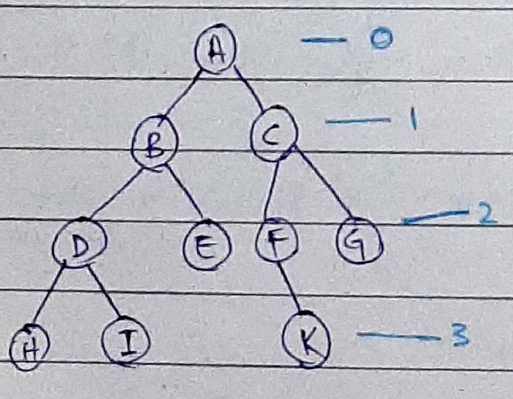
→ Iterative Deepening Depth First Search

→ Combination of DFS & BFS.

→ Finds out best depth limit & increases limit gradually until goal is found.

→ BFS: fast search & DFS: memory efficient.

→ Useful when: large search space & depth of goal node is unknown.



A

A → B → C

A → B → D → E → C → F → G

A → B → D → H → I → E → C → F → K → G

Complete: Yes (if B finite)

Time: $O(b^d)$

Space: $O(bd)$

Optimal: Yes (if cost = 1)

Dazzle

Day / Date: ____________

→ Informed Search

→ Search Heuristics

Algos have additional info along with goal state, which helps in efficient searching.

• Heuristic

→ func that estimates how close a state is to a goal.

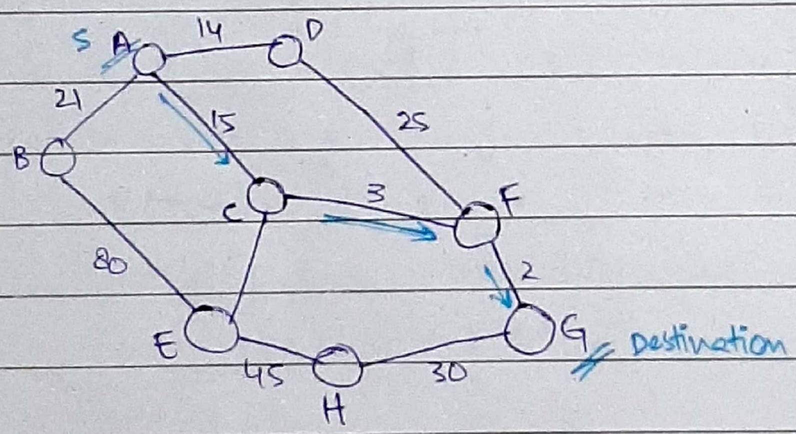
→ Designed for a particular search prob

→

→ GREEDY BEST FIRST SEARCH

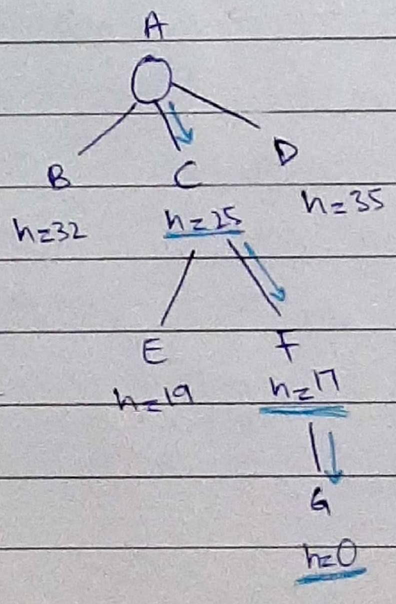
→ expand closest node to the goal node. The 'closeness' is estimated by heuristic $h(x)$.

~~→ explored~~ → The ~~cost~~ heuristic will be considered & explore node by node.



$h(x)$

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



Path from A→G:

$\boxed{A \to C \to F \to G}$

Time comp = $O(b^d)$ → efficient

but doesn't give optimal sol.

Dazzle

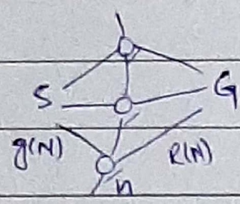
Day / Date: ____________

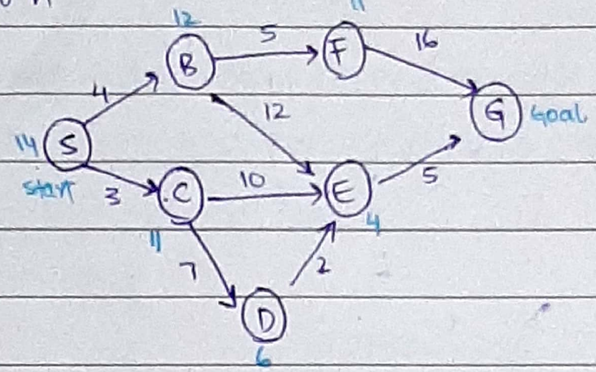
→ A* Algorithm

$f(N) = g(N) + h(N)$ → Estimation cost from n to goal node

↓

Actual cost from start to n





→ Always optimal sol.

$f(S) = 0 + 14 = 14$

$S \to B$

$4 + 12$

$= 16$

$S \to C$

$3 + 11 = \underline{14}$

↓

$SC \to E$

g(N) $3 + 10 + 4$

$= 17$

$SC \to D$

$\underbrace{3+7}_{} + 6$

$= 16$

Since both have similar cost we can explore

~~SDE~~ $SCD \to E$

$3+7+2+4$

$= 16$

$SB \to F$

$= \underbrace{4+5}_{} + 11$

$= 20$

$SB \to E$

$= \underbrace{4+12}_{} + 4$

$= 20$

less than 20 so explore SDE more.

path: SCDEG

cost = 17

$O(b^d)$ → complexity space + time

$SCDE \to G$

$= 3+7+2+5+0$ heuristic of goal is always 0.

$\boxed{= 17}$

Day/ Date:

GAME THEORY

Adversial Search

- Competitive & Cooperative Search — A search where we examine the problem ~~ix~~ which arises when we try to plan ahead of world & other agents are planning against us.

↓ search in which two or more players with conflicting goals are trying to explore same search space for the sol is called Adversial Search, often known as Games.

Game Theory → branch of maths used to model strategic interaction b/w different players in context with a predefined ~~se~~ rules & outcomes.

* Games are modeled as a search prob & evaluation func that helps to solve games.

Perfect info — A game with perfect info is one in which agents can look into complete board.

Imperfect info — Agents donot have all info about the game.

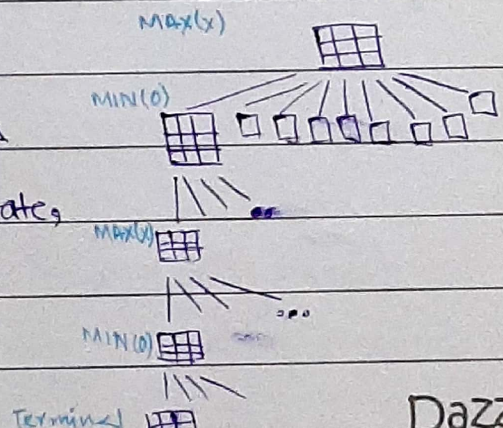
Deterministic games — Follows a strict pattern or rules for game.

Non-deterministic games — Various unpredictable ~~game~~, events.

- ZERO-SUM GAMES — Adversial search which ~~allows~~ involves pure competition.
  - → Each agent's gain/loss of utility is balanced by gains/losses of another agent.
  - → One player tries to maximize one single val. while other tries to minimize it.

- GENERAL GAMES — Agents have independant utilities.
  - → Cooperative, indifference.

- GAME TREE — A tree where nodes of tree are game states & edges are moves by players. Game tree involves initial state, actions func and result func (utility).



Dazzle

Day/ Date:

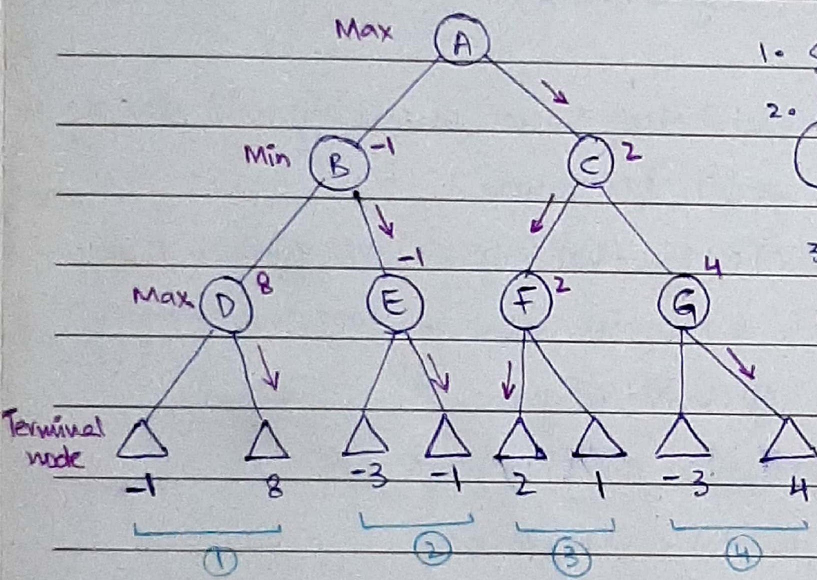
→ MIN - MAX ALGORITHM

→ Two players play the game — one is MIN & other is MAX; best move strategy.

→ Both players fight it as opponent gets minimum benefit while the other get max benefit.

→ Max will try to maximize its utility (best move), min will try to minimize utility (worst move)

→ Min-max algo perform depth first search algo for exploration of tree; Backtracking algo



1. → compare values of ① first & then ②
2. → compare values of ③ & then ④
→ find max.
3. find min for B & min for C.
4. find max at A.

→ ∴ if player follows path A → C → F, it will win with utility = 2.

Time complexity = $O(b^d)$ & Space complexity ; complete — will definitely find a sol (if exist)

→ Min-max algo proceeds all the way down to the terminal node of the tree & then backtracks the tree as recursion.

Algo is optimal if both players are playing optimally.

Dazzle

Day/ Date:

→ ALPHA - BETA PRUNING

→ An optimization technique for min - max algo.

→ $\alpha - \beta$ cuts off branches in the game tree which need not to be searched b/c there already exists a better move available.

Two parameters:

α – value of best choice (highest val.) we've found so far along the path for MAX. $(-\infty)$

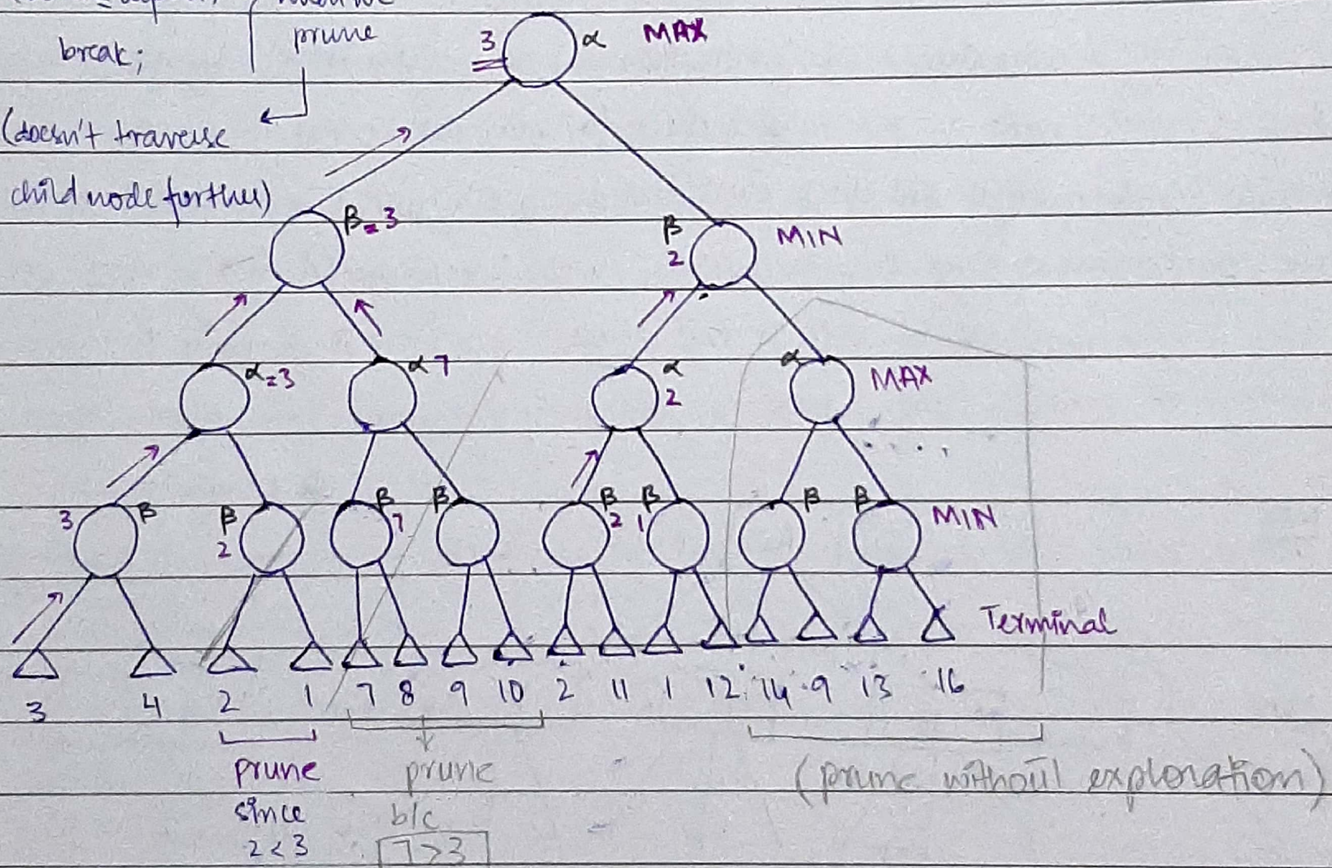
β – value of the best choice (lowest val) we've found so far along the path for MIN. (∞)

$(\alpha \geq \beta)$ Best value: child node se traverse krke best value kiya bn hai parent ki.

if (beta ≤ alpha) } when we prune

break;

(doesn't traverse child node further)



Time complexity: $O(b^{d/2})$

Advantage → less traversing of nodes

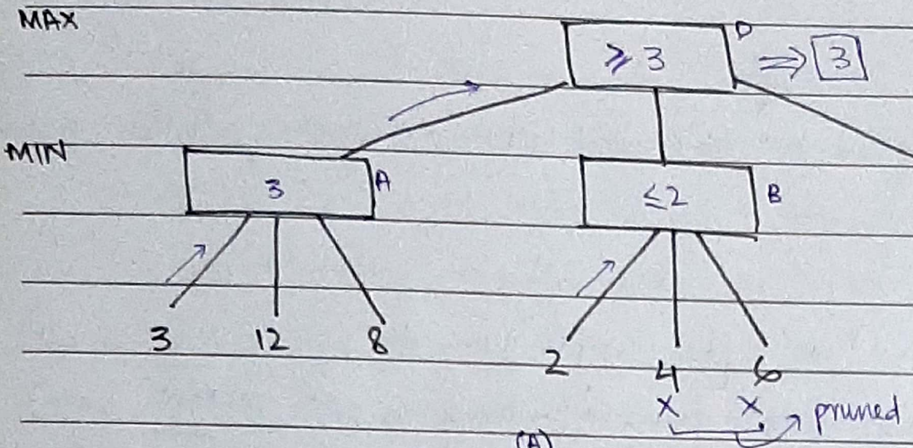
Drawback → Totally dependant on values (mainly order/value of child nodes)

* If search space is made on domain knowledge, it gives better results (higher chances of succeed) & leaf nodes ordering might be better resulting in more pruning).

Dazzle

Day/ Date:

Short way



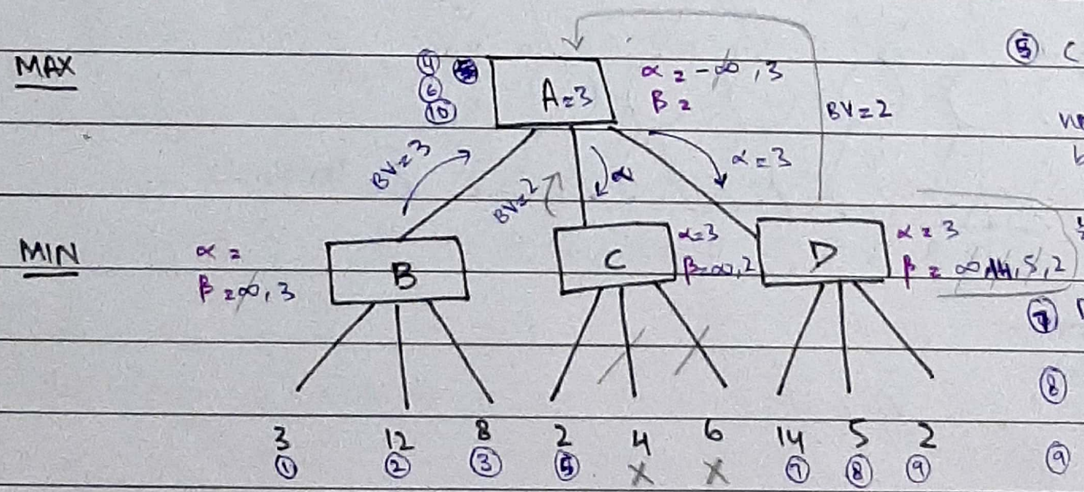
→ first we traverse first child (A), pehli value 3 read ki & baqi se compare krke min nikal liya.

→ Now next child val. should be ≥3 (cause max) only then it will be final node. Assume parent node (D) as ≥ 3 value & check for next child's (B) val.

→ Since 2 < 3 ∴ we'll not check other childs of B & prune branches 4 & 6.

→ check node values of C → 14 ≥ 3 consider it. check next child 5 ≥ 3 & 2 ≤ 3. since 2 min most val consider it BUT since 3 ≥ 2 ∴ 3 is final D value.

X ———— X



⑤ $C = min(\infty, 2)$
$= 2$
now since $\beta \leq \alpha$ so break & donot traverse further child.
& return BV to parent

⑦ $D = min(\infty, 14)$
$= 14$

⑧ $D = min(14, 5)$
$= 5$

⑨ $D = min(5, 2) = 2$

① $B = min(\infty, 3) = 3$

② $B = min(3, 12) = 3$

③ $B = min(3, 8) = \boxed{3} = \alpha$

3 is our best value (BV)

④ $A \to \alpha \to max(-\infty, 3)$
$\alpha = 3$ (temporary)

⑥ $A = \alpha = max(3, 2)$
$\alpha = 3$

→ child node parent ko best value detay hain (pass on)

→ parent node child ko α, β value jo calculate ki hai wo pass karta

⑩ $A \to \alpha = max(3, 2)$
$= \boxed{3} \longrightarrow$ ans

Dazzle

3/21/3

(lec 15)

CSP – CONSTRAINT SATISFACTION PROBLEM

Why?

can solve prob in less-time. If constraints are pre-defined, gives effective results.

> States are represented independently. & what values are allocated to them.

→ States are considered as seperate entities. CSP reduces search space.

Base of CSP–

① Variable: Seperate entities (eg. in n Queens, $Q_1, Q_2 \dots Q_n$ are variables).

Something jisko hum value assign krein aur sol. nikalein; assign location.

② Domain: what options have been provided regardless of the constraints; set of options

eg. for Q_1 The options are (0,0) (1,0) (2,0) (3,0).

③ Constraints: constraints < scope, relation > are the limitation/restriction.

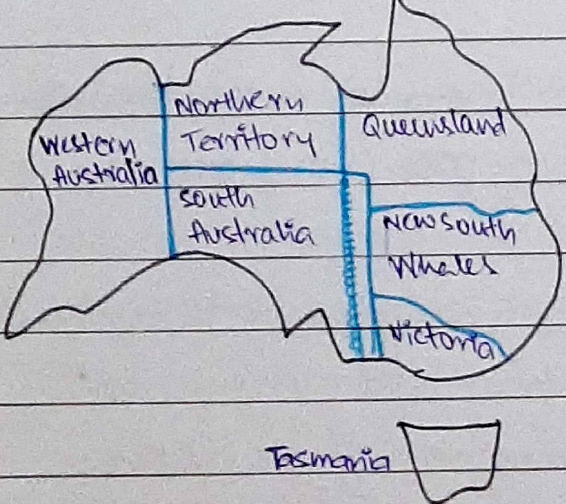
$Q_1 \rightarrow \times Q_2$ → Q_2 cannot be vertical, horizontal or diagonal to Q_1.

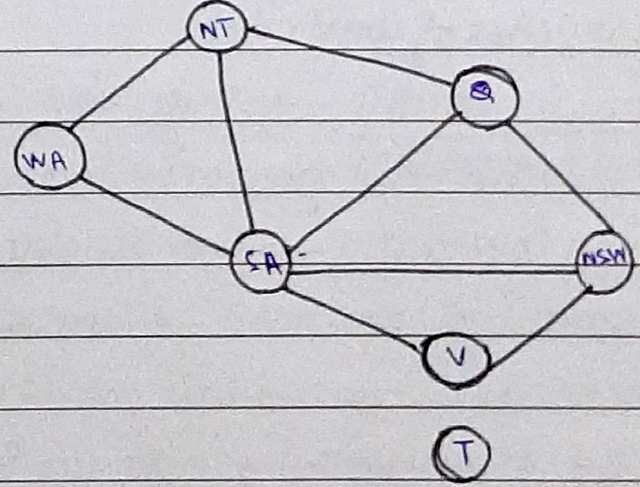
$\downarrow \quad \searrow \times Q_2$

Q_3

Example probs → N-Queen prob, Map coloring, Job Scheduling, Crypt Arithmetic

MAP COLORING → Color in such a way that no two neighbours have same colour.





Variables: {NT, WA, SA, Q, NSW, V, T}

Domain: {R, G, B} → colours provided

Constraints: no two neigh. should have same colour or {WA ≠ NT, WA ≠ SA ..}

Day / Date: ______ artificial

→ JOB SCHEDULING
- Memory managers in OS (FIFO, Round Robin, LRU).
- Pipelining in CA.

Example → Scheduling assembly in a car. (in slides)
→ axles have to be in place before the wheels are put on & it takes 10 mins to install an axle → $\boxed{\text{constraint: Axle} + 10 \leq \text{wheels}}$
→ another → affix + 1 ≤ nuts
nuts + 2 ≤ hubcap
→ we need to assert that the inspection comes last and takes 3 min.
All other tasks + $\underbrace{d}_{\text{time limit}}$ ≤ inspection.
→ eg. to get whole assembly done ~~by~~ in 30 min = means all tasks takes 27 mins

Infinite Domains — DK time limit. Cannot give values for comparison.
Time during is taking as an arbitary variable. • Use constraint language
> linear programming is a good eg. for infinite domain (constraints to solve inequal)

~~Vari~~ Varieties of Constraints
→ Unary Constraint → Single variable is involved to define constraint.
eg. WA doesnot want green color in map coloring prob (WA ≠ green)
→ Binary Constraint → Two variables : WA ≠ TA
→ Preference (soft) constraint → Especially preferred a particular constraint for a better sol. (hojaye tou achi baat, na bhi hou tou sahi hai); NOT necessary to apply.
eg. Timetable → class timing preferences || Map coloring → NT prefers Red color.
→ High order constraint → Has arbitary no. of ~~const~~ variable. Can be less but variable can be of a large no. eg. crypthasithmic puzzle.

Day/ Date: ____________

→ Cryptarithmetic Puzzle (Real life encryption & decryption)

* digits represents a letter. Each letter is represent by a unique ~~no~~ digit.

Goal → find the digits such that a given mathematical eq. is verified.

- Variables = {a ... z} → the alphabets being used.
- Domain = {0 ... 9}

Example:

$$
\begin{array}{cccc}
c_3 & c_2 & c_1 & \\
 & T & W & O \\
+ & T & W & O \\
\hline
F & O & U & R
\end{array}
$$

V = {O, W, T, F, U, R}

D = {0 – 9}

C = {values of alphabet should have unique digits}
(no two alphabet can have same digit)

F = {1} b/c carry cannot be greater than 1; carry always 1

assume
eg $O + O = R + (10 \times c_1)$
$9 + 9 \quad 8 + (10 \times 1)$
$18 = 18$

constraint 1: $O + O = R + 10(c_1)$

constraint 2: $c_1 + W + W = U + 10c_2$

*we don't know if c_1, c_2 exist.

constraint 3: $c_2 + T + T = O + 10c_3$

constraint 4: $c_3 = F$

$$
\begin{array}{ccccc}
 & c_2 & & & \\
 & \times & & & \\
 & 7T & W & O\,4 & \\
1 + & 7T & W & O\,4 & \\
\hline
F & O & U & R & \\
\downarrow & \downarrow & \downarrow & \downarrow & \\
1 & 4 & 6 & 8 &
\end{array}
$$

first assume O = 4.
trial & error seems ayega.

→ Cryptarith. has a restriction that digits have to be from 0 – 9 (10 digits) only.

→ Example of CSP (example of timetable from slides)

Variable → one variable per class, so V = {C_1, C_2, C_3, C_4, C_5}

Domain → Professors → D = {X, Y, Z}

Qs. For each variable (class) define its constraints.

C_1 = {Z}
C_2 = {Z, Y}
C_3 = {Z, Y, X}
C_4 = {Z, Y, X}
C_5 = {Z, Y}

domains of each variable
↓
extracted from already given data in ques

Day/ Date:

The classes are:

C1 = PF 8 – 8:55
C2 = AI 8.30 – 9:25
C3 = NLP 9 – 9:55
C4 = Robotics 9 – 9:55
C5 = ML

① (C1 ≠ C2) → binary constraint — C1 can not be taught by same prof as C2
② C2 ≠ C3 , ③ C2 ≠ C4
④ C3 ≠ C4 , ⑤ C3 ≠ C5
⑥ C4 ≠ C5

Sol → • C_1 → Prof Z
• C_2 → Prof Y cause $C_1 \neq C_2$
and since C2 ≠ C3 & ≠ C4 ∴ Y is removed from C2 & C3
• C3 → Prof X b/c C3 ≠ C4 & C3 ≠ C5 so C4 only have option Z.
• C4 → Prof Z
& since C4 ≠ C5 so,
• C5 → Prof Y.

> Shrinking (domain) options with help of constraints to find ideal sol. in less time.
→ while shrinking domain if any domain of some variable become ϕ null so we know that sol. is wrong & we backtrack.

LOCAL CONSISTENCY → approaches of constraint propogation.

→ CONSTRAINT PROPOGATION

* If we have constraints, how can we shrink our domain; when & Domain, variables in it & The constraints are given.

⇒ ① Node consistency

> Equivalent to considering unary constraints of a particular variable.
→ If values in the variable's domain satisfy the variable's unary constraint.

Day/Date:

Eg. if state SA = {R, G, B} → domain in map colouring prob & SA ≠ Green is a constraint given, so we'll shrink our domain → SA = {R, B} to satisfy ~~node~~ unary constraint.

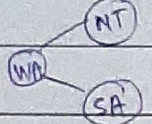
⇒ particular node is consistent according to its unary constraints & has domain restricted acc to it.

⇒ ② Arc consistency

→ considering binary constraints acc to the nodes its directly connected with.

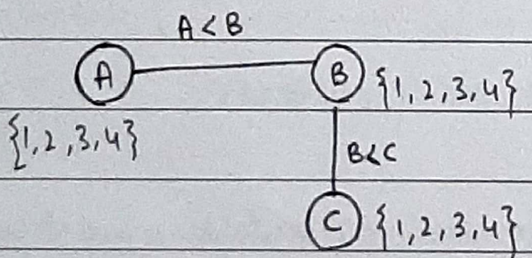
> If every value in variable's domain satisfies the variable's binary constraints.

Eg



→ WA mein koi aisi value nai rakh saktay jiski waja se NT aur SA consistent na hown WA ke sath.

→ AC-3 Algo is used for Arc consistency.



→ A domain of range of numbers is given to each node with predefined conditions (A<B & B<C) (constraints)

→ Maintain an arc queue

→ Pop an arc one by one

①

~~A-B~~	~~B-A~~	~~B-C~~	C-B

① (A<B) A — B

A {1,2,3,4} → new domain = {1,2,3}; B {1,2,3,4}

(Arc from A to B) so,

- make A consistent with respect to B
- for every value of A, there should be a value in B which satisfies the constraint of A.

so we shrink domain of A to {1,2,3}

important condition)

{ → If domain of A is shrinked, check if it is connected with any other node whose arc isn't added up in the queue & add it. }

② B —(B>A)— A ; make B consistent wrt A → New domain of B = {2,3,4} ('check this condition')

{1,2,3,4} {1,2,3}

③ B —(B<C)— C → Domain of B shrinked to {2,3} → check condition & add A-B in queue.

{2,3,4} {1,2,3,4}

(add child node first jo B ka domain shrink honay ki waja se affect hou)

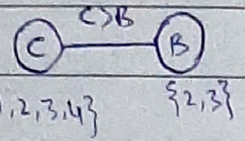
Dazzle

Day/ Date:

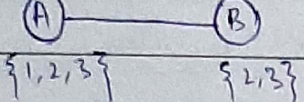
Updated Queue. | A ~~→B~~ | B ~~→A~~ | B ~~→C~~ | C-B | A - B |

→ new added

↳ x ↳

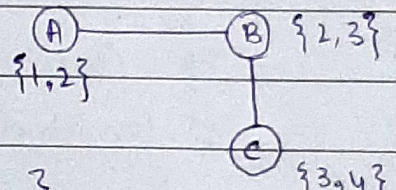
④ 

→ Domain of C shrinks to {3, 4} → check condition. (false)

⑤ 

→ Domain of A = {1, 2}

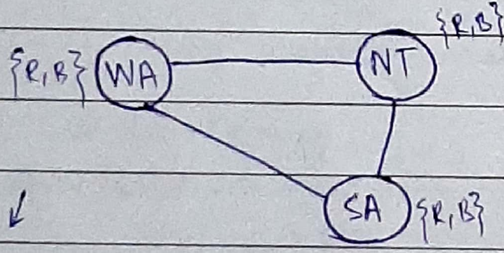
→ Queue empty & final domains are:



{ Satisfies constraint, acc to rules of arc consistency. } of variables

⇒③ Path Consistency

In which we have a middle node that should necessarily considered.



when considering 2 variables, we should assign those v such values that they satisfy the constraint of the node that is in the middle of path.

acc. to arc consistency, {R,B} satisfies constraints of each other, but if we want to give colours to $\boxed{WA}$ & $\boxed{SA}$ such that constraints of {WA, NT} & {NT, SA} are also satisfied, then there is no possible sol. for this graph with 2 colours given.

(WA — NT) & (NT — SA)

Red — ☐ ☐ — Blue

Blue — ☐ ☐ — Red

} we need a third colour for NT to satisfy constraints.

→ Aka 2 arc consistency & generalized form.

Note: • constraints are also imposed on variables, not domain.
• Domain is shrinked using those constraints.

Day/ Date:

⇒ ④ k consistency

> k is variable so it is a generalized consistency.

> if k=1 ⟶ node con. , k=2 ⟶ arc , k=3 ⟶ path & so on.

if k=4 ⟶ given 3 variables such values that the 4th variable can also stay consistent w.r.t to other 3 variable since they are interlinked.

⇒ ⑤ Global constraints

> Arbitrary no. of variables. Ok - the no. of variables or constraints.

Example ⟶ Sudoku

Variables ? (9x9 grid) ⟶ so 81 variables

Domain ? {1 – 9}

constraints ? ① In each row, all unique digits. ② In each column, all unique digits
③ In each 3x3 box, all unique digits.

	1	2	3	4	5	6	7	8	9
A			3		2		6		
B	9			3		5			1
C			1	1		6	4		
D			6	8		2	9		
E	7					$\boxed{4}$			8
F			8	7		8	2		
G			2	6		9	5		
H	8			2		3			9
I			5		1	$\boxed{7}$	3		

⟶ choose the most constrainted box so that the value can be found out easily.

① ⟶ E6 : 1, 2, 7, 8, 5, 6, 9, 3 ⟶ only 4 left.

② ⟶ I6 : 1, 2, 3, 6, 9, 4, 5, 8, ⟶ Only ~~8~~ 7 left

* look at constraints & keep figuring out options.

⟶ Real life example of constraint propogation

⟶ Resources to be allocated in OS.

Day/ Date:

BOUNDED PROPOGATION (limiting the range)

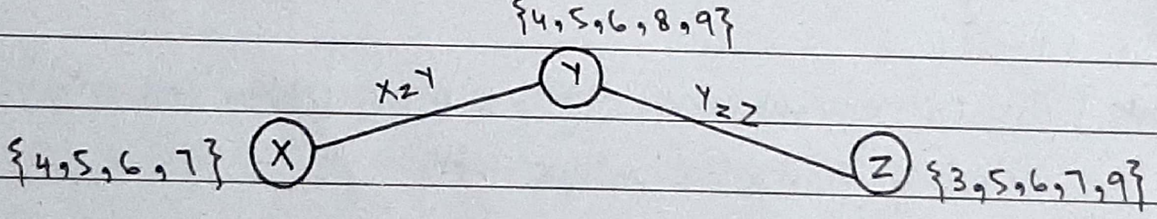
Two flights F_1 & F_2 with the range of passangers [0,165] & [0,385], respectively.
$F_1 + F_2 = 420$ ⟶ This condition should be fulfilled.

F_1	F_2
[0,165]	[0,385]
↓	↓

now if we want max passengers in either of them, then

[35, 165]	[255, 385]	⟶ converted in this form using constraint $F_1 + F_2 = 420$

ARC Consistency example 2



Queue

| ~~X-Y~~ | Y-X | Y-Z | Z-Y |

① (X) —X≥Y— (Y) ⟶ New domain of X = {4,5,6}
{4,5,6,7} {4,5,6,8,9} ⟶ check condition of neighbouring child being affected

② (Y) —— (X) ⟶ Domain of Y = {4,5,6}
{4,5,6,8,9} {4,5,6}

③ (Y) —— (Z) ⟶ Domain of Y = {5,6}, check condition since domain Y shrunk.
{4,5,6} {3,5,6,7,9}
So, X is a neighbour.
therefore add X-Y in queue

Dazzle

Day/ Date:

new added

Updated Queue : | ~~X-Y~~ | ~~Y-X~~ | ~~Y-Z~~ | Z-Y | X-Y |

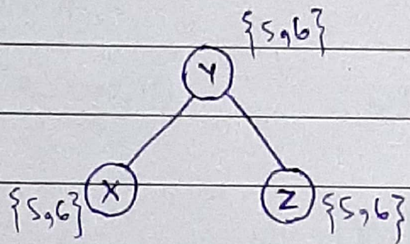
④ Z —— Y → Domain of Z = {5, 6} ; condition of add-on fail

{3, 5, 6, 7, 9}   {5, 6}

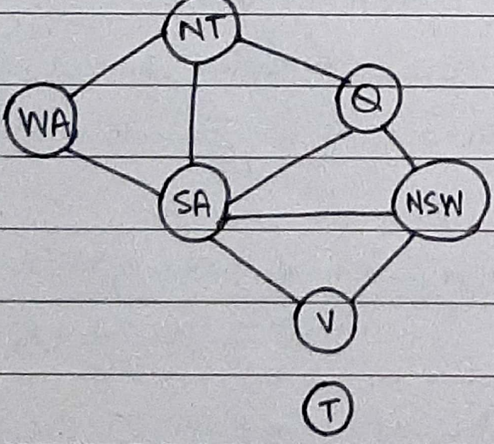
⑤ X —— Y → Domain of X = {5, 6} ; condition of add-on fail.

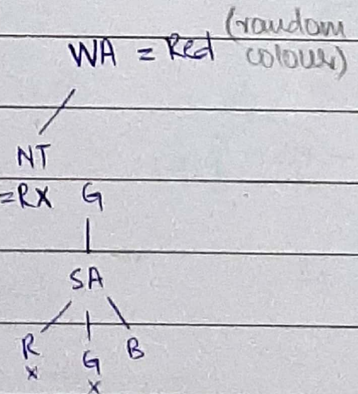
{4, 5, 6}   {5, 6}

Final result →



BACKTRACKING





If normal backtrack, assign each state some colour, check & move forward or backtrack.

→ Time consuming, lengthy & complex.

- less complex & easier values

① MRV — Minimum Remaining Value (Best & leading Approach)
(most constrainted variable)

→ to choose state which has minimum val. remaining ; order is maintained in this manner.

(i) Variable Ordering → which ~~next~~ variable should be given priority next (acc to some approach)

(ii) Value Ordering → After a variable chosen, which value should be given preference.

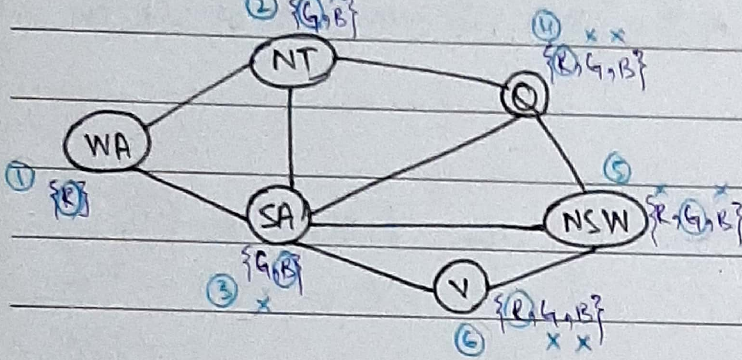
→ follow Variable Ordering ONLY!

Dazzle

→ Drawback of MRV : Initial node nai pata.
→ MRV & Degree Heurestic gives sol. quickly.

Day/ Date:

(Takay jis variable ke pas minimum values ie. colors bachay hain usko pehlay assign krdein takay us variable ka domain zero na hojaye)



↓ (first variable is choosen randomly)

① If WA ko R dein tou us ke connecting neighbours' domain will shrink. Baqi ki same rahay gi.

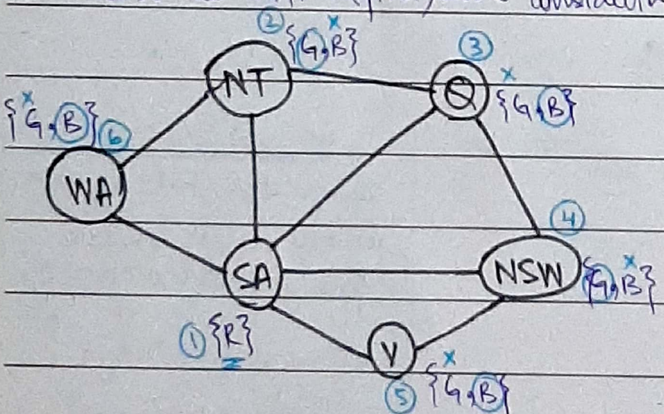
② Choose G for NT ; domains for Q & SA will shrink.

③ Choose B for SA ; domains shrink for others

④ Q = Red ; ⑤ NSW = Green ; ⑥ V = Red.

② Degree Heurestic (Variable Ordering)

Choose the initial (first) node considering the ~~highest~~ degree ; Degree — in / out going arcs



① Choose SA first b/c connected to 5 nodes (↑ degree) & assign any (Red) colour to it.

② NT has degree 3 ; shrinking domains of others

→ Go in the order of highest → lowest degree.

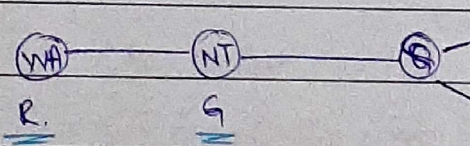
Highest degree → maximum no. of variables. (most constrainted).

× ——— ×

Value Ordering → value ko preference dein ; least constraining value is chosen.

→ Choose a value that least shrinks the domain of other. Doesn't empty / zero domain of others.

Eg:



→ Red (if given so fine)

→ Blue (if given so domain of SA will become empty Therefore choose red)

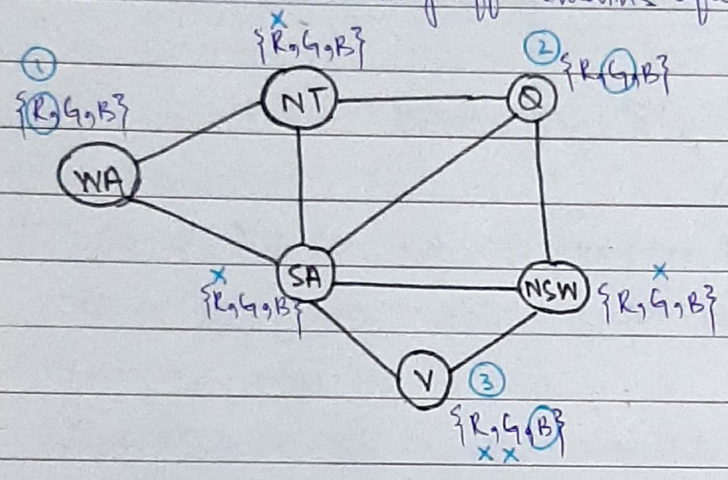
→ (apna bhi faida aur doosray ka bhi)

Day/ Date: ____________

③ → Forward Check

Arc consistency → pre processing step (value assign krnay se pehlay hei domain shrink krdetay)

Forward Check → cutting off domains after assigning values to nodes.



→ Either Arc consisentency is needed or forward check. Both are NEVER required.

→ WA = Red
Q = Green
V = Blue
} pre defined colours to be assigned.

→ when we assign Blue to V, SA will have empty domain. This is the drawback.

→ Although the issue arised when we assigned Q = Green but forward didnot tell & assigned further. So, drawback : only tells we need to backtrack when a node's domain becomes empty.

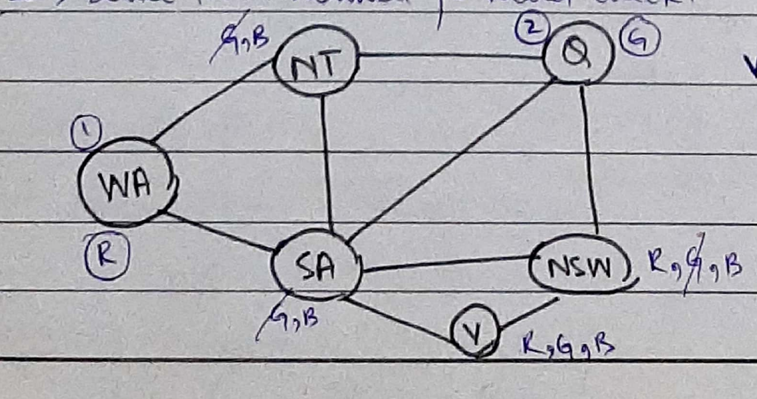
→ The ~~domains~~ nodes cancel out the domains of eachother regardless of the fact that the nodes are becoming inconsistent with each other.

④ → Improvement in forward check (through heurestic approach)

Can be done through applying MRV or degree heurestic.

⑤ → MAC (Maintaining Arc Consistency) {forward check + arc consistency}

→ Better than normal forward check.



WA = Red
Q = Green
V = Blue

→ Along with applying forward check, keeps applying arc cons. (AC-3 algo) to check if the neighbouring nodes are satisfied.

Day/ Date:

Consider the neighbours → NT, SA & NSW. → domains of These shrinked out

→ check if neighbours are affected, so put in queue.

SA — NT	WA — NT	WA — SA	NT — SA	V — SA	NSW — SA	SA — NSW	V — NSW

SA — NT → failure ; domain empty before queue ended.

B B

→ so at This point only we'll change colour of Q. before assigning colour to V.

useful : If we want to assign colours and choose nodes randomly, so its better that we apply AC-3 algo along to keep checking for inconsistency rather than getting to know in end & backtracking whole map.

⑥ Backjumping (only know concept) (modified version)

→ Better version of normal backtracking, but useless as compared to forward check & others.

→ Rather than going back to immediate parent node, directly go to The node which has affected the answer. & emptied the node.

Eg. {Q = Red, NSW = green, V = blue, T = red} & Order = {Q, NSW, V, T, SA, WA, NT}

④ T = Red.

Q = Red

WA = ~~{G, B}~~

NT = {G, B} {Q,

SA = {G, B} ① ③ {Q, NSW, V}

NSW = {G, B} {Q,

V = {G, B, R} ② {NSW

① Q = Red
|
NSW = Green.
|
V = Blue
|
T = Red
|
SA → since SA domain is empty, we'll backtrack to last parent node of SA ie. V & change colour of that value going back to immediate parent ie. T

→ Maintain a linked list ke kis ki waja se domain shrink kiya hai. Add parent node in linked list along with shrinking domain.

Day/ Date: ______

→ Practice Question of CSP Problem

variant of 4 house prob. There are 4 families A, B, C, D living in 4 houses – 1, 2, 3, 4.

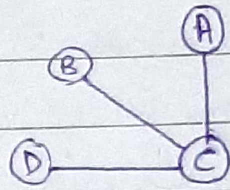
① C lives in a house with higher no. than D $(C > D)$

② D lives in a house next to A in a lower no. house. $(D+1 = A)$

③ There is atleast one house b/w D & B. $(|D-B| \geq 2)$

④ C doesnot live in 3. $C \neq 3$

⑤ B doesnot live in 1. $B \neq 1$ — unary constraints.



Solve using Arc consistency.

→ Initial Domains

$A = \{1, 2, 3, 4\}$ →③ $\{2,3,4\}$ →⑦ $\{2,3\}$

$B = \{1, 2, 3, 4\}$ → $\{2, 3, 4\}$ →⑤ $\{3,4\}$

$C = \{1, 2, 3, 4\}$ → $\{1, 2, 4\}$ →① $\{2, 4\}$

$D = \{1, 2, 3, 4\}$ →② $\{1, 2, 3\}$ →⑥ $\{1, 2\}$

(additional add up)

Queue : | ① C–D | ② D–C | ③ A–D | ④ D–A | ⑤ B–D | ⑥ D–B | ⑦ A–D |

① C —($C>D$)— D
$\{\cancel{1}, 2, 4\}$ $\{1, 2, 3, 4\}$

② D —($D<C$)— C
$\{1, 2, 3, \cancel{4}\}$ $\{2, 4\}$

③ A —($A-1=D$)— D
$\{\cancel{1}, 2, 3, 4\}$ $\{1, 2, 3\}$

④ D —($D+1=A$)— A → no domain shrinked
$\{1, 2, 3\}$ $\{2, 3, 4\}$

⑤ B —($|D-B| \geq 2$)— D
$\{\cancel{2}, 3, 4\}$ $\{1, 2, 3\}$

⑥ D — B
$\{1, 2, \cancel{3}\}$ $\{3, 4\}$

⑦ A — B
$\{2, 3, 4\}$ $\{1, 2\}$

$A = \{2, 3\}$
$B = \{3, 4\}$ ⇒
$C = \{2, 4\}$
$D = \{1, 2\}$
↓
shrinked domains

for solution

If $A = 2$
↓ use MRV
$C = 4$
↓
$B = 3$
↓
$D = 1$

Dazzle

Day/Date: 4. April. 20

PROBABILITY

→ Mostly used in machine learning b/c AI works on probability ; Concept of Prob:

① Decision Theory

Provides a formal framework for making logical choices in the face of uncertainity & risks considering ~~##~~ a set of alternatives, consequences and correspondence b/w those sets.

→ Basic decision is based on probability + utility → (The decision gives max profit.)

* maximize excepted performance based on decision.

BASIC NOTIONS

→ (i) sum of event = P(1)

→ (ii) if P(a) = 0.5 so P(!=a) = 1 − P(a)

→ (iii) Prior Prob: * conditional prob → based on evidence ; unconditional → vice versa

(P(a|b) → prob of a given b)

P(a), P(c) = —

simple prob given.

"Given that" condition not provided.

① $P(a|b) = \frac{P(a,b)}{P(b)}$ or $\frac{P(a \wedge b)}{P(b)}$

② P(a or b) = P(a) + P(b) − P(a ∧ b)

this is also formula of P(a∪B)

→ (iv) Rule of Independence

If P(a|b) are independent of eachother, so P(a∧b) = P(a) * P(b) b/c they have no common point.

so, $P(a|b) = \frac{P(a \wedge b)}{P(b)} = \frac{P(a) \times P(b)}{P(b)}$ $\boxed{= P(a)}$

∴ $\boxed{P(a|b) = P(a)}$ if a & b are independent of eachother

Day/Date:

Joint Probability Distribution
(* multiples no of options of each) ; toothache & ¬, cavity & ¬, catch & ¬ so 2x2x2 = 8 options

Example 1

	toothache		¬ toothache	
	catch	¬ catch	catch	¬ catch
cavity	0.108	0.012	0.072	0.008
¬ cavity	0.016	0.064	0.144	0.576

① P(cavity) = ? = 0.108 + 0.012 + 0.072 + 0.008

② P(toothache) = ? = 0.108 + 0.012 + 0.016 + 0.064

③ P(toothache or cavity) = ? P(a) + P(b) − P(a ∧ b) = P(toothache) + P(cavity) − (0.108 − 0.012)

④ P(cavity | toothache) = ? $\frac{P(\text{cavity} \wedge \text{toothache})}{P(\text{toothache})}$ $= \frac{0.108 + 0.012}{0.108 + 0.012 + 0.016 + 0.064}$

⑤ P(¬ cavity | toothache) = ? 1 − P(¬ cavity | toothache) (b/c given that same evidence)

⑥ Normalization constant in conditional probability
constant term that is normalizing the val. & bringing it = 1

eg. $P(c|t) = \frac{P(c \wedge t)}{P(t)}$ & $P(\neg c|t) = \frac{P(\neg c \wedge t)}{P(t)}$

so here, P(t) is normalization constant which % value & bring sum = 1.

BAYES RULE (Imp for classification model prob)

$$P(a|b) = \frac{P(b|a) \times P(a)}{P(b)}$$

eg. $P(\text{cause}|\text{effect}) = \frac{P(\text{effect}|\text{cause}) \times P(\text{cause})}{P(\text{effect})}$

cause → result/label
effect → evidence/symptom

Day/Date: ______________________

Example 2 (Doctor Example from book)

Data:

$P(s) = 0.01$

$P(m) = 1/50000$

$\Big\}$ prior prob.

$P(s|m) = 0.7$ (disease m causes patient to have a stiff neck 70% of the time)

– Find $(m|s)P = ?$

$$P(m|s) = \frac{P(s|m) \times P(m)}{P(s)} = \frac{0.7 \times 1/50000}{0.01}$$

Example 3 (Past Paper)

$P(m) = 1/1000$

$P(f|m) = 0.9$ (if people have malaria then they can have high fever)

$P(f|\neg m) = 0.15$

Find:

(A) If you have high fever, what is the prob that you have malaria?

$$P(m|f) = \frac{P(f|m) * P(m)}{P(f)}$$

	m	¬m
f		
¬f		

$$P(m|f) = \frac{0.9 \times 1/1000}{P(f)}$$

where $P(f) = P(f \wedge m) + P(f \wedge \neg m)$

$$P(f|m) = \frac{P(f \wedge m)}{P(m)} \quad ; \quad P(f|\neg m) = \frac{P(f \wedge \neg m)}{P(\neg m)}$$

$P(f \wedge m) = P(f|m) \times P(m)$ | $P(f|\neg m) \times P(\neg m)$

substitute here

$$\left(0.9 \times \frac{1}{1000}\right) + \left(0.15 \times 1 - \frac{1}{1000}\right)$$

$P(f) = \boxed{}$

(B) Prob that you have malaria but donot have fever?

$$P(m|\neg f) = \frac{P(\neg f|m) \times P(m)}{P(\neg f)}$$

$\Rightarrow$ since $P(f|m) = 0.9$ $\therefore$ $P(\neg f|m) = 0.1$

$P(\neg f) = 1 - P(f)$

Day/ Date: 5. April. 20

CLASSIFICATION

→ Supervised Learning

→ Past info with label is given, so we train our model acc. to it & classify new data.

eg • decision tree, regression

→ Unsupervised Learning

→ Dataset is provided, but labels not given on any. We have to take decisions on basis of features & info without labels. eg. clustering, frequent item set mining, dimensionality reduction.

→ Reinforcement Learning

Take a decision to give max performance based on the environment you're thrown into. If past data not given, ~~then~~ give exposure to model. eg. self driving car. initial decision may be wrong → will get eventually get better as it takes exposure.

NAIVE BAYES CLASSIFIER → features always INDEPEDANT of each other, classifier assumes (∴ easy to solve & give acc. ans)

→ Discrete based Naive Bayes classifier

A discrete set of features.

1 Eg: Given a training set with color, type & origin as features & Stolen (Y/N) as label.

Find: (table given in Classification 2 video)

Given Red, SUV & domestic find whether it would be classified as stolen or not

Bayes: $P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$

$P(stolen = yes \mid RSD) = \frac{P(R,S,D \mid stolen) \times P(stolen)}{P(R,S,D)}$

where, $[P(R,S,D \mid stolen) \times P(stolen)] + [P(R,S,D \mid \neg stolen) \times P(\neg stolen)] = P(R,S,D)$ b/c ← normalizing constant

also, * $P(stolen = Y \mid R,S,D) + P(\neg stolen \mid RSD) = 1$

$P(stolen) = 5/10$, $P(\neg stolen) = 5/10$.

since R, S, D are independant ∴ $P(R,S,D \mid stolen) = P(R \mid stolen) \times P(S \mid stolen) \times P(D \mid stolen)$

$\boxed{(P(R|stolen) \to stolen = 5, \text{ unmein se } Red = 3)}$ $= \frac{3}{5} \times \frac{1}{5} \times \frac{2}{5}$

P(α | given) sample

$\boxed{P(R,S,D \mid stolen) = 6/125 = 0.048}$

Dazzle

Day/ Date: ____________

$P(R, S, D \mid \neg stolen) = P(R \mid \neg stolen) \times P(S \mid \neg stolen) \times P(D \mid \neg stolen)$

$= \frac{2}{5} \times \frac{3}{5} \times \frac{3}{5}$

$\boxed{P(R, S, D \mid \neg stolen) = 0.144}$

$P(stolen = Y \mid R, S, D) = \frac{0.048 \times 5/10}{P(R, S, D)}$; $P(stolen = N \mid R, S, D) = \frac{0.144 \times 5/10}{P(R, S, D)}$

$= \frac{0.024}{P(R, S, D)}$; $\frac{0.074}{P(R, S, D)}$; $P(R, S, D) = 0.024 + 0.074$

$= 0.25$; $\boxed{= 0.75}$; $\boxed{P(R, S, D) = 0.096}$

Since $P(stolen = NO \mid R, S, D)$ ∴ car will not be stolen.
as per training dataset

→ for ans, choose the one with high prob.

2 Eg: Given refund = no & martial status = single, find whether cheat = Yes or No.

Refund	Martial Status	Cheat
Yes	Single	NO
NO	Married	N
N	S	N
Y	M	N
N	Divorced	Yes
N	M	NO
Y	D	NO
N	S	Y
N	M	N
N	S	Y
NO	SINGLE	? → find

$P(cheat) = 3/10$; $P(\neg cheat) = 7/10$

$P(cheat \mid R = No, M = S) = \frac{P(R = No, M = S \mid ch) \times P(cheat)}{P(R = NO, M = S)}$

$P(\neg cheat \mid R = NO, M = S) = \frac{P(R = NO, M = S \mid \neg ch) \times P(\neg cheat)}{P(R = NO, M = S)}$

$P(R = NO, M = S \mid cheat) = P(R = No \mid cheat) \times P(M = S \mid cheat)$
$= 3/3 \times 2/3 = 0.66$

$P(R = NO, M = S \mid \neg cheat) = P(R = No \mid \neg cheat) \times P(M = S \mid \neg cheat)$
$= 4/7 \times 2/7 = 0.16$

$P(cheat \mid R = No, M = S) = 0.66 \times 0.3 = 0.198$

$P(\neg cheat \mid R = No, M = S) = 0.16 \times 0.7 = 0.112$

$\boxed{\therefore \text{will cheat} = \text{YES}}$

Day/ Date: 11.April.20 (Classification-3)

Naive Bayes with Continuous Variables
&
Results Evaluation Parameters

Gaussian Naive Baye's Rule → Used for real valued attributes; continuous val. with no fixed range & assumes features are independent of one another & values are normally distributed.
(Takes in consideration mean & variance/standard deviation & work with it to give output)

→ Training dataset – divide dataset in training & testing. Training data should be more for better performance & maximum learning – (70% – training & 30% – testing) – division

Step 1: This is not a requirement (just an extra work done to improve accuracy)
↓ feature scaling (bring values of features in range [0,1])
(to decrease diversity)
formula: $\dfrac{\text{value at a particular index} - \text{that column's min}}{\text{that column's max} - \text{that column's min}} \Rightarrow \boxed{\dfrac{x - min}{max - min}}$ → normalization factor

Step 2:
(i) Count no. of classes → No. of classes = 2
(ii) Prior probability calculate of classes → P(class 0) = 4/7 , P(class 1) = 3/7

Step 3:
Mean, variance & Standard deviation (extra)
→ calculate mean & variance for all the rows that belong to class 0 & 1 for all features seperately.

* classify test data.: after feature scaling:

f_1	f_2	f_3	f_4
1	0.669658	0.974864	0.667718

2nd row

take val from testing data → x_i; mean → μ; variance → σ^2

$$\frac{e^{-\left(\frac{x_i - \mu}{2\sigma^2}\right)}}{\sqrt{2\pi\sigma^2}}$$

formula for gaussian NB

Dazzle

Day/ Date:

→ NOW FOR LABELLING TESTING DATA

→ we've taken 2^{nd} row from testing data for practicing (video classification 3)

f_1	f_2	f_3	f_4
1	0.669658	0.974864	0.667718

calculate for each class:

① $P(class=0 | f_1, f_2, f_3, f_4) = \dfrac{P(f_1 f_2 f_3 f_4 | class=0) \times P(class=0)}{P(f_1, f_2, f_3, f_4)}$ 4/7

→ donot calculate b/c sum of numer. = denominator

$P(f_1, f_2, f_3, f_4 | class=0) = P(f_1|class=0) \times P(f_2|class=0) \times P(f_3|class=0) \times P(f_4|class=0)$

$P(f_1|class=0)$ | $P(f_2|class=0)$ | $P(f_3|class=0)$ | $P(f_4|class=0)$

$= \dfrac{e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}}$

$= \dfrac{e^{-\frac{(1-0.962806)^2}{2\times0.001671}}}{\sqrt{2\pi * 0.001671}}$

$= \boxed{6.45132889}$

$= \dfrac{e^{-\frac{(0.669658-0.922945)^2}{2\times0.002453}}}{\sqrt{2\pi\times0.002453}}$

$= \boxed{1.686\times10^{-5}}$

$= \dfrac{e^{-\frac{(0.97486-0.954)^2}{2\times0.002231}}}{\sqrt{2\pi\times0.00231}}$

$\boxed{= 7.7101}$

$= \dfrac{e^{-\frac{(0.667718-0.410)^2}{2\times0.121062}}}{\sqrt{2\pi\times0.121068}}$

$\boxed{= 0.87220}$

$$P(class=0|f_1 f_2 f_3 f_4) = \frac{[6.45132889 \times 1.686\times10^{-5} \times 7.7101 \times 0.87220] \times [4/7]}{P(f_1, f_2, f_3, f_4)}$$

= 4.1767e-04

NOTE

* while calculating for $P(f_1 f_2 f_3 f_4|class=0)$, if any val = 0, the whole val = 0 b/c multiplication.

② $P(class=1|f_1, f_2, f_3, f_4) = \dfrac{P(f_1 f_2 f_3 f_4|class=1) \times P(class=1)}{P(f_1, f_2, f_3, f_4)}$

Result

	Prob. of class = 0 wrt each row	Prob. of class = 1 wrt each row
2^{nd} →	4.1767 e-04	0

→ choose max of both col for each row to classify class = 1 or 0.

→ 2^{nd} row belong to class 0.

Dazzle

Day / Date: 17. April. 20

PERFORMANCE EVALUATION MEASURES

RESULTS EVALUATION

There are certain parameters which enables us to deduce that how good a particular model has been for a dataset ~~(is/absolutely good enough/or not)~~

1. Accuracy (how much % is it correct) (100% acc if no false)
2. Confusion matrix → errors → correct values predicted

		Actual outcome T	F
Predicted Outcome	T	T +ve TP	F -ve FP
	F	F -ve FN	T -ve TN

T +ve → We predicted as T & it is actually also T
T -ve → We predicted as false & it is actually also F
(correct values (100% acc.))

F +ve → We predicted T however it was not (Type I error) (type I error)

F -ve → We predicted F however it was not (Type I error) (type II error)

→ Accuracy = (TP + TN) / (TP + TN + FP + FN) { how many correctly predicted out of total

→ 3. Specificity = TN / TN + FP (aka true -ve rate) { jitnay ghalat honay thay usmein se kitnay correctly -ve mark kiye hain

→ 4. Sensitivity = TP / TP + FN (aka True +ve rate or recall) { jitnay true honay thay usmein se kitnay correctly predict kiye

→ 5. Precision = TP / TP + FN { how many are true out of actual true that you've predicted

eg. Qs:

		Actual class Cat	Dog	Rabbit	
Predicted class	Cat	5	2	0	7
	Dog	3	3	2	
	Rabbit	0	1	11	
		8			

found correctly cat

found correctly ¬cat

→ construct a confusion matrix for class cat.

Sol:

predicted	Actual cat	¬cat
cat	5	2
¬cat	3	17

→ find intersection point for each ~~class~~ options.

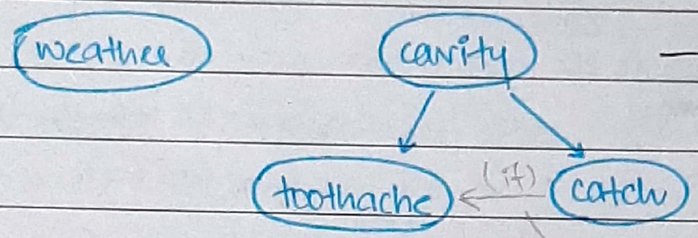
→ find confusion matrix for each class & then calculate parameters & add them up to evaluate result.

Day/ Date: 19. April. 20

BAYESIAN NETWORK

→ Why Study? How to remove flaw of Naive Bayes ie. if features are not independent.
* Correlation used; told beforehand. (dependent/not)

Example 1:



— parent

— child

P(toothache | cavity, catch)

* child is dependant on parent.
* If there's an edge b/w node then dependency occurs else not!

In this example:

→ Weather & cavity are independent

→ toothache & catch are dependant on cavity.

If we are asked to calculate:

- P(weather, cavity, toothache, catch) = ?

It would be equal to: P(weather) x P(cavity) x P(toothache | cavity) x P(catch | cavity)

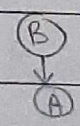
used b/c dependency

In general, we some node X is dependent, we write: P(X | all parent_nodes)

Example 2: P(A, B, C) = ?

if: ① Independant? ⇒ P(A) x P(B) x P(C)

② B is parent of A? ⇒ P(A | B) x P(B) x P(C)



→ cavity, toothache, catch & weather are variables

→ The arrows/edges are the network showing correlation b/w variables.
and these form a network which might have probabilities.

Day/ Date:

NETWORK TOPOLOGY

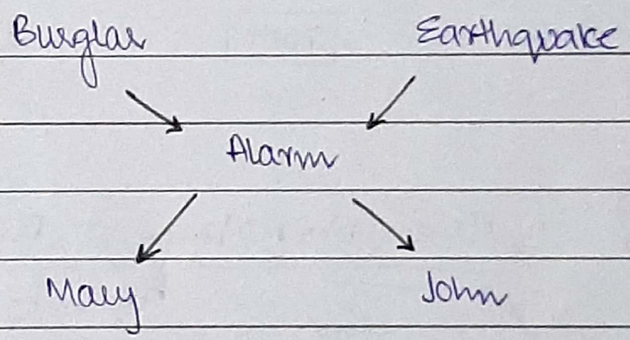
Example: I am at work. Neighbour John calls to say my alarm is ringing, but neighbour Mary doesn't call. Sometimes, it is set off by minor earthquakes. Is there a burglar?

Variables → John calls, Alarm, Mary, Earthquake, Burglar

Note: (i) Mary & John's call is dependant on the alarm. If alarm rings → then only they might call, else they will not call at all.

(ii) The alarm will only ring if there are some mishaps i.e. earthquake or burglar or else the alarm will not go off; Alarm is dependant on mishaps.

Network →

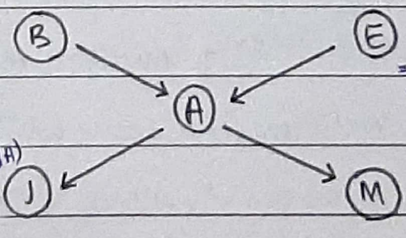


$$P(J, M, B, A, E) = P(J|A) \times P(M|A) \times P(B) \times P(A|B,E) \times P(E)$$

Example:

$P(\neg B)$	$P(B)$
$1-0.001 = 0.999$	0.001

$P(\neg E)$	$P(E)$
$=1-0.002 = 0.998$	0.002



B	E	$P(A \mid B,E)$	$P(\neg A \mid B,E)$
T	T	0.95	
T	F	0.94	
F	T	0.29	
F	F	0.001	

A	$P(J \mid A)$	$P(\neg J \mid A)$
T	0.90	
F	0.50	

A	$P(M \mid A)$
T	0.70
F	0.01

* Black pointer wali prob. are given. Blue is sol.

(Ques mein parent ki jo bhi prob. poochi hou, sol. mein wohi 'given' mein likhein ge hamesha)

Q. $P(J \wedge M \wedge A \wedge \neg B \wedge \neg E) = ?$

$= P(J|A) \times P(M|A) \times P(A|\neg B, \neg E) \times P(\neg B) \times P(\neg E)$

$= 0.90 \times 0.70 \times 0.001 \times (1-0.001) \times (1-0.002)$

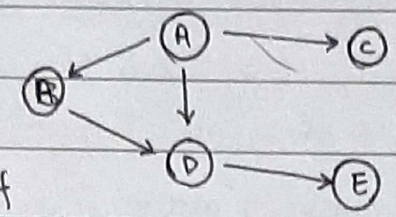
$\boxed{= 6.281 \times 10^{-4}}$

Day/ Date: ______________

Example:

→ Each variable can be either on or off.

- $P(A=on) = 0.6$
- $P(B=on|A) = \begin{cases} 0.1 & A=off \\ 0.95 & A=on \end{cases}$
- $P(C=on|A) = \begin{cases} 0.8 & A=off \\ 0.5 & A=on \end{cases}$



- $P(D=on|A,B) = \begin{cases} 0.1 & A=off, B=off \\ 0.9 & A=on, B=off \\ 0.3 & A=off, B=on \\ 0.95 & A=on, B=on \end{cases}$
- $P(E=on|D) = \begin{cases} 0.8 & D=off \\ 0.1 & D=on \end{cases}$

Find:

① $P(A=on, B=on, C=on, D=on, E=on) = ?$

$= P(A=on) \times P(B=on|A=on) \times P(C=on|A=on) \times P(D=on|A=on, \overset{B=on}{}) \times P(E=on|D=on)$

$= 0.6 \times 0.95 \times 0.5 \times 0.95 \times 0.1 = \boxed{0.027}$

② $P(E=on|A=on) = ?$

additional prob. of B.

$= \dfrac{P(E=on, A=on)}{P(A=on)} = \dfrac{\underbrace{P(E=on|D)}_{\text{given}} \times P(D|A=on, B) \times P(A=on) \times P(B|A=on)}{P(A=on)}$

$= P(E=on|D) \times P(D|A=on, B) \times P(B|A=on)$

If parent not in ques, then we'll calculate its separate probability, and consider both on & off condition (all possible conditions).

If given parent already in ques then we'll not add it separately. If condition of a variable given i.e. on or off, we'll consider same while expanding formula.

→ since we dk exact case therefore we'll consider all combinations & OR them. i.e. add them all

	D	B	P(E=on|D)		P(D|A=on, B)		P(B|A=on)
1.	off	on	0.8	×	0.05	×	0.95
2.	on	off	0.1	×	0.9	×	0.05
3.	on	on	0.1	×	0.95	×	0.95
4.	off	off	0.8	×	0.1	×	0.05

$(0.8 \times 0.05 \times 0.95) + (0.1 \times 0.9 \times 0.05) + (0.1 \times 0.95 \times 0.95) + (0.8 \times 0.1 \times 0.05) = \boxed{0.13675}$

Dazzle

Day/ Date:

③ $P(A=on | E=on) = ?$ Acc. to Baye's rule $\rightarrow P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$

$= \frac{P(E=on | A=on) \times P(A=on)}{P(E=on)}$ ~ since $P(E=on | A=on) \rightarrow$ solved in ques 2
∴ directly put value.

$= \frac{0.13675 \times 0.6}{P(E=on)}$

for $\rightarrow P(E=on) = P(E=on | D) \times \underbrace{P(D | A, B) \times P(A) \times P(B | A)}$

since ~~prob~~ condition unknown, we'll make table of all possible combos.

A	B	D	P(E=on|D)	P(D|A,B)	P(A)	P(B|A)	
on	on	off	0.8	× 0.05	× 0.6	× 0.95	0.0228
on	off	on	0.1	0.9	0.6	0.05	2.7×10^{-3}
on	on	on	0.1	0.95	0.6	0.95	0.0545
on	off	off	0.8	0.1	0.6	0.05	2.4×10^{-3}
off	on	off	0.8	0.7	0.4	0.1	0.0224
off	off	on	0.1	0.1	0.4	0.9	3.6×10^{-3}
off	on	on	0.1	0.3	0.4	0.1	1.2×10^{-3}
off	off	off	0.8	0.9	0.4	0.9	0.2592

$= (0.8 \times 0.05 \times 0.6 \times 0.95) + (0.1 \times 0.9 \times 0.6 \times 0.05) + \ldots + (0.8 \times 0.9 \times 0.4 \times 0.9)$

$= 0.36845$

putting in ①

$= \frac{0.13675 \times 0.6}{0.36845} = \boxed{0.2226}$

Day/ Date: 24. April. 20

CLUSTERING

→ Unsupervised learning approach, no labels. → learning from observation rather than training dataset.

Applications:

① Web ② Customers (cluster people acc to same/similar interest

③ Image pattern (recognizing digits by seeing different images with different writing).

④ Outlier Detection (noise — something happen which doesn't frequently happen)

⑤ Document clustering — doc with similarity ; ⑥ ~~Doc~~ Topic modelling.
(topic ke hisab se model krna)

→ Euclidean - Manhattan distance is used in clustering

Euclidean — $\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$

Manhattan — $|x_2-x_1| + |y_2-y_1|$

TYPES: ① Partitioning
(k-means) ← only this in course
· Kmediod
· Kmodes

② Hierarchical (BIRCH algo)
to make a tree, hierarchy & link up similar elements through it.

③ Density Based.
gives a threshold how many elements, points & dataset.
→ (Bottom up approach acc. to similarity - clustering one by one)

K - MEAN

Example: (where A & B are coordinates for 7 rows)

Subject	A	B
1	1	1
2	1.5	2
3	3	4
4	5	7
5	3.5	5
6	4.5	5
7	3.5	4.5

→ k = 2.

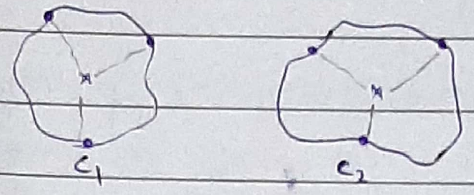
Initial seeds → Jin k coordinates ki basis pe cluster krtay hein & it's equal to k given. If k=2 then 2 seeds

* we choose randomly

Dazzle

Day/ Date:

SSE – Sum of squared error.



→ har point ka centroid se distance & then sq. them & summing them all ; x → centroid representation

→ If better clustering → SSE will be less.

→ SSE should be min. after every iteration.

→ Do clustering until you reach point of convergence.

when current & previous iteration has no difference. means no further iteration needed.

Sol : We chose (1,1) & (5,7) as initial seeds (randomly) & then we need to calculate euclidean distance of every point from these 2 initial seeds.

Iteration 1:

	Seed 1 (1,1)	Seed 2 (5,7)
1	0 ✓	7.21
2	1.12 ✓	6.10
3	3.61 ✓	3.61
4	7.21	0 ✓
5	4.72	2.5 ✓
6	5.32	2.06 ✓
7	4.30	2.92 ✓

Iteration 2:

	Seed 1 (1.83, 2.33)	Seed 2 (4.125, 5.375)
1	1.57	5.37
2	0.467	4.275
3	2.039	1.77
4	5.64	1.85
5	3.15	0.73
6	3.78	0.53
7	2.74	1.08

$\boxed{(1,2)\ C_1 \qquad (3,4,5,6,7)\ C_2}$

Since C_1 & C_2 of iteration 1 & 2 are not same ∴ we'll do 3rd iteration. The answer of 3rd itr. will be same so we'll stop. that will be our final ans.

Euclidean with ②: (1.5, 2)

S1

$= \sqrt{(1.5-1)^2 + (2-1)^2}$

$= 1.12$

S2

$= \sqrt{(1.5-5)^2 + (2-7)^2}$

$= 6.10$

→ The seed with minimum ~~cluster~~ euclidean, the point will go to that cluster (seed).

→ Result of Iteration 1: (1,2,3) C1 & (4,5,6,7) C2.

calculate centroid of coordinates of both cluster's points

(1,1) (1.5,2) (3,4)

$\frac{1+1.5+3}{3}, \frac{1+2+4}{3}$

$\boxed{(1.83, 2.33) \quad (4.125, 5.375)}$

new seeds for iteration 2.

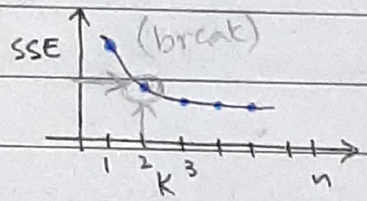
Dazzle

Day/ Date: ____________

(not in course)

→ If not mentioned in ques, what is the best no. of clusters (k)?

- plot a graph b/w SSE & clusters from (1 . . . no. of docs)
- calculate at each no. of cluster (k) the ~~dist~~ SSE (distances from centroid ka square) & plot it on graph
- The graph will ALWAYS have a BREAKING POINT. That point is our best k value.



→ DISADVANTAGE OF KMEAN (part of course)

→ It is ONLY for numeric dataset.

→ Very sensitive to outliers (noise) → It highly affects the mean (Agar koi bohot chotti ya bari val. ho)

→ Makes spherical clusters (always) so there can be a chance that it forms unnatural clusters

→ Not in course

→ k-mediod : uses median to form clusters & choose seeds

→ k-mode : uses modes to form clusters & choose seeds

↓ both are expensive approaches b/c point of convergence comes after many iterations. But it is not much affected by outliers.

Silhouette coefficient → an approach to check if clustering is done correctly.

formula: $\frac{b-a}{\max(b,a)}$, values ranges from [-1, 1]

If: 1 → best clustering , -1 → worst clustering.

↓ works on phenomenon of seperation & cohesion

seperation: how far you are from elements in other clusters

cohesion → how closer you are with elements in your cluster.

$\boxed{\text{cohesion > seperation}}$ → always

b → har point ka min. distance from other clusters

a → distance of each point from other points of its own cluster.

Day / Date: 3. May. 20

DECISION TREES

→ classification model — gives format in tree structure.

→ strategy to tell which attributes are the best for you to decide label — attribute priority in tree format.

• Handles missing data in test data — mostly used when there are errors or missing values.

→ Also called inductive inference approach b/c we ~~to~~ see which attribute is giving most info of dataset.

→ ID3 algo is used.

Problems

→ works on discrete dataset. (Discrete range of datavalues) — can work on continuous dataset but we have to convert val. in discrete range.

∘ Important terms

• Entropy — dataset is partitioned into subsets of similar values — measures homogenity of your samples.

• Info gain (IG) — calculated for each attribute — tells how much one attribute individually is giving info about the dataset.
(more ig → more imp) ; root node is most imp

→ entropy uses IG.

Question is in lec video of (Decision Tree I)

Sol

Entropy $= \sum_{i=1}^{classes} \left(- P_i \log_2 (P_i) \right)$

Yes	No	Total
9+	5-	14

$\underbrace{-\frac{9}{14} \log_2 \left(\frac{9}{14}\right)}_{Yes} - \underbrace{\frac{5}{14} \log_2 \left(\frac{5}{14}\right)}_{No} = \boxed{0.940}$

Entropy of class label

① Step 1 : Entropy of your label

② Step 2 : calculate IG of each attribute & choose highest one for the root.

IG(attr.) = entropy(parent) − [weighted avg * entropy(child)]

Dazzle

Day/ Date:

~~IMP~~ !

* If there is no node in tree ∴ parent node will be class label initially. entropy of → first time calculation

(i) IG (outlook) = $0.940 - \left[\frac{5}{14} \times 0.970 + \frac{4}{14} \times 0 + \frac{5}{14} \times 0.970\right]$ $= \boxed{0.247}$ → max. val.

	Sunny	Overcast	Rain	
weighted avg →	5	4	5	→ no. of each in ques table
	[2+, 3−]	[4+, 0−]	[3+, 2−]	→ how many of each belongs to yes & no.
	$-\frac{2}{5}\log_2\left(\frac{2}{5}\right) - \frac{3}{5}\log_2\left(\frac{3}{5}\right)$	$-\frac{4}{4}\log_2\left(\frac{4}{4}\right) - \frac{0}{4}\log_2\left(\frac{0}{4}\right)$	$-\frac{3}{5}\log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right)$	
	$= \boxed{0.970}$	$\boxed{= 0}$	$\boxed{= 0.970}$	

Entropy is always in range of [0 − 1] → If attribute belongs to same class, entropy = 0
If attribute is equally divided [2+, 2−], entropy = 1

(ii) IG (Temperature) = $0.940 - \left[\frac{4}{14} \times 1 + \frac{6}{14} \times 0.918 + \frac{4}{14} \times 0.811\right]$ $= \boxed{0.029}$

Hot	Mild	Cool
4	6	4
[2+, −2]	[4+, 2−]	[3+, 1−]
e = 1	$-\frac{4}{6}\log_2\left(\frac{4}{6}\right) - \frac{2}{6}\log_2\left(\frac{2}{6}\right)$	e = 0.811
	e = 0.918	

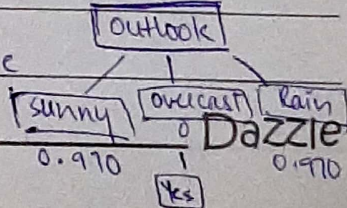
(iii) IG (humidity) = $0.940 - \left[\frac{7}{14} \times 0.985 + \frac{6}{14} \times 0.5\right]$ $= \boxed{0.151}$

High	Normal
7	7
[3+, 4−]	[6+, 1−]
e = 0.985	e = 0.5916

(iv) IG (wind) = $0.940 - \left[\frac{8}{14} \times 0.811 + \frac{6}{14} \times 1\right]$ $= \boxed{0.048}$

weak	strong
8	6
[6+, 2−]	[3+, 3−]
e = 0.811	e = 1

→ since highest IG is of outlook attribute ∴ it will be root node



Day/ Date:

Note : we'll stop making tree when leaf nodes of a tree become class label.
whenever we have $e = 0$, we'll classify that node into label b/c the type belongs to one specific class, so we end that ~~leaf~~ branch & make a leaf node as label.
→ we've classified overcast as YES in first iteration.

(2) → Now calculate IG of temp, humidity & wind.

(i) $IG(temp) = 0.970 - \left[\frac{2}{5} \times 0 + \frac{2}{5} \times 1 + \frac{1}{5} \times 0\right] = \boxed{0.57}$

(e of sunny b/c parent)

Hot	Mild	Cool
2	2	1
[0+, 2-]	[1+, 1-]	[1+, 0-]
e = 0	e = 1	e = 0

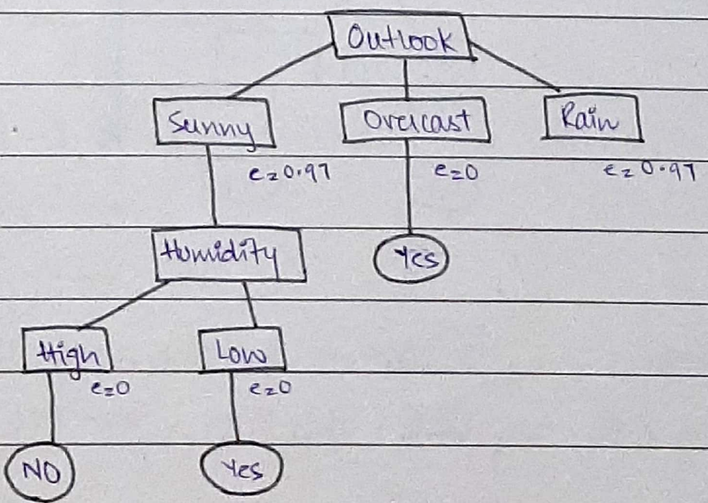
→ (consider only sample of sunny ie 5)

(ii) $IG(humidity) = 0.970 - \left[\frac{3}{5} \times 0 + \frac{2}{5} \times 0\right] = \boxed{0.970}$ → max. value so most imp

High	Low
3	
[0+, 3-]	[2+, 0-]
e = 0	e = 0

(iii) $IG(wind) = 0.970 - \left[\frac{2}{5} \times 1 + \frac{3}{5} \times 0.918\right] = \boxed{0.0192}$

Strong	weak
2	3



Day / Date:

③ Now calculate for rain as parent (5 samples, $e = 0.970$)

(i) $IG(temp) = 0.970 - [0 + 3/5 \times 0.918 + 2/5 \times 1] = \boxed{0.0192}$

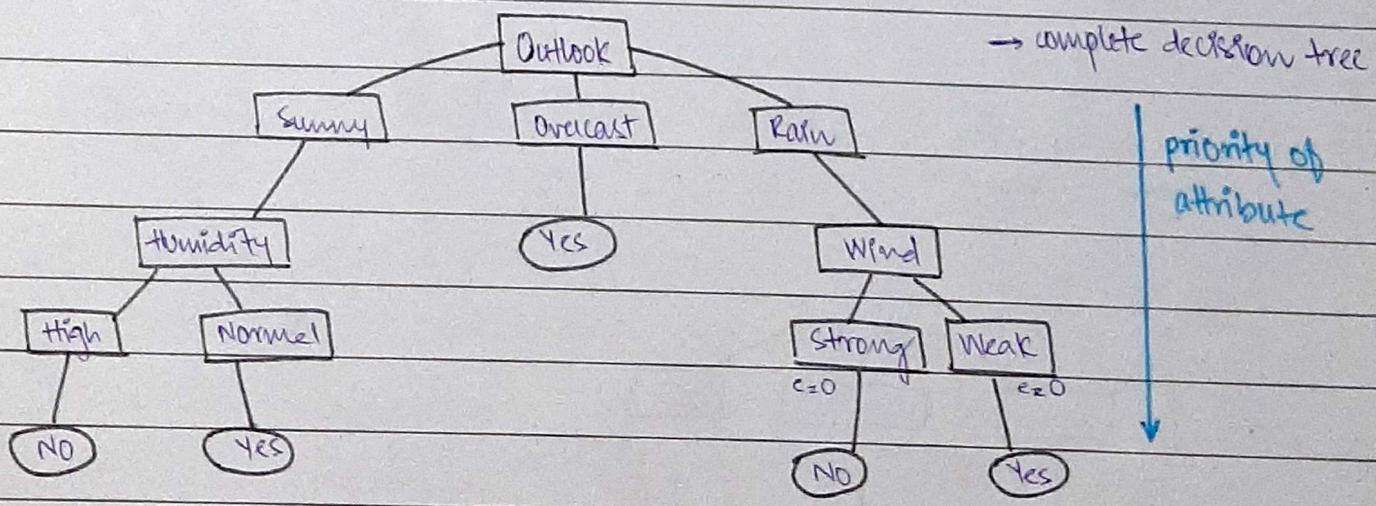
Hot	Mild	Cool
0	3	
[/]	[2+, 1-]	[1+, 1-]

(ii) $IG(humidity) = 0.970 - [\frac{2}{5} \times 1 + \frac{3}{5} \times 0.918] = \boxed{0.0192}$

High	Normal
2	
[1+, 1-]	[2+, 1-]
$e = 1$	$e = 0.918$

(iii) $IG(wind) = 0.970 - [2/5 \times 0 + 3/5 \times 0] = \boxed{0.970}$ → max val.

Strong	Weak
2	3
[0+, 2-]	[3+, 0-]
$e = 0$	$e = 0$



→ now if we have a testing data — outlook = sunny, temp = high, humidity = high, wind = ? (missing val)
we can easily classify it as 'NO' using 2 attr. only (~~sunny~~ outlook & humidity).

Day/ Date: 11. May. 20

(Last topic)

LINEAR REGRESSION

→ Supervised learning, more like classification (use past to predict future / the odds)

* If 2 variables, so 1 independent & 1 dependant.

independant

$Y = mx + c$ → intercept

dependant, gradient

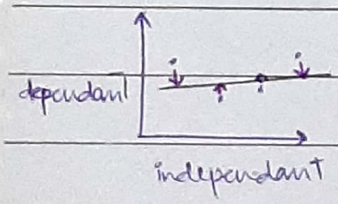
m & c are calculated

→ The eq. is made by based on training dataset & then we test it on testing dataset.

↓ based on training data we predict dependant variable ki value kiya aaay ka chance hai.

- used in economic, health, stock & financial dept — risk analysis, health/disease analysis

→ Shown using Scatter Diag. (use R lang.)



method of least sq. (residual)

→ make a linear line on this graph that has minimum distance from every point.

(sab ke distance ke sq. ka summation)

→ to have minimal error.

Example: Training dataset → (1,2), (2,1), (4,3). ; $y = mx + c$

① mean(x) → $\frac{1+2+4}{3}$

② mean(y) → $(2+1+3)/3$

③ mean(xy) → $(1\times2)+(2\times1)+(4\times3)/3$

④ mean(x^2) → $(1^2+2^2+4^2)/3$

} we need to cal. these 4 values for m

x ——— x

End.

Dazzle

Day/ Date:

gradient (m) $= \dfrac{\text{mean}(x) \cdot \text{mean}(y) - \text{mean}(xy)}{(\text{mean}(x^2))^2 - \text{mean}(x^2)}$ — put values

$= 0.428$

intercept (c) $= \text{mean}(y) - (\text{gradient} * \text{mean}(x))$

$= 1.01 \approx 1$

forming eq. by $y = mx + c$

$$\boxed{y = 0.428x + 1}$$

now we can easily classify testing data using this eq.

correlation coefficient (R)

→ tells how much the variables x & y are correlated ; value ranges from $[-1, 1]$

-1 → negatively corelated (aik increase tou doosra decrease) 

$+1$ → positively corelated (aik increase tou doosra bhi increase) 

0 → variables are not corelated at all.