1) Uninformed & informed search

state space -> set of all possible states

successor function -> outputs successor state and the cost of action given action

start state, goal test

state space graph or search tree

Uninformed search

maintain a fringe, to expand (following strategy) remove element -> place its children in fringe

DFS -> not complete (bcz of cycles)
not optimal (find leftmost not best)

BFS -> complete
not optimal (only if edge cost are all equivalent)

UCS -> expand node with least cost from starting position (backward cost)
complete
optimal if all costs >0

Informed search

Heuristics -> estimate of reach of goal

must be a lower bound to actual cost of reaching goal (optimistic)
for being admissable an heuristic must be non negative and optimistic
for being consistent $\forall A, C \quad h(A) - h(C) \leq cost(A, C)$ A, C nodes in the graph
consistency => admissability

Greedy search -> dont use backward cost but forward cost

not complete and not optimal

A* search -> combine forward and backword cost

complete and optimal (given admissable heuristics)

2) CSP

identify if state is goal, defined by:
- variables -> can only have 1 value from a set
- Domain -> set of values a variable can take
- Constraints -> restrictions on the values of the variables (and relation between them)

can be defined as search problems (state -> partial assignments)

constraints can be represented through a graph

backtracking search -> fixed ordering, select values for variables
only select values that don't conflict, if no val. available backtrack and change val.

forward checking -> when selecting value prune conflicting values of variables that share constraints

Arc consistency -> arc in both directions for each edge

check $X_i \to X_j$ for each value of x_i checked if there is one in X_j that doesn't violate constr.

if not remove the value. If removed something have to check all nodes that share

constraint with X_i

the order in which I pick the variables can have an impact on performanc

Minimum remaining values -> select the variable that most likely causes a back track

Least constraining value -> select the value that generates the least problems

3) Adversial search

agents have adversaries. Can be deterministic or stochastic
Adversial search returns a strategy/policy that raccomends the best move given config. of agents and adv.

Minimax → assume adv. plays optimally for them.
can create a game tree. A state value is the best possible outcome (utility) obtainable from the state
Agents alternate, one picks max value from children, the other min

alpha-beta pruning → if I'm looking for max above and in one of the below (min) I find something lower than max prune (for sure lower so no need to check it)

evaluation function → estimate the maximax of a state, used for deput limited minimax
each feature has a weight

expectimax → introduce chance nodes which consider the avg case (expected value)

4) Probability

conditional probability : $P(x|y) = \frac{P(x,y)}{P(y)}$

Product rule : $P(x,y) = P(x|y)P(y)$

chain rule: $P(x_1,..x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1,x_2)... = \prod_{i=1}^{n} P(x_i|x_1,..x_{i-1})$

X, Y indidendent iff : $\forall x,y: P(x,y) = P(x)P(y)$

X and Y are conditionally indipendent given Z iff : $\forall x,y,z: P(x,y|z) = P(x|z)P(y|z)$
$(X \perp\!\!\!\perp Y | Z)$

5) Bayes' Nets

Query variables → unknown, left side

Evidence variables → Known value, right side

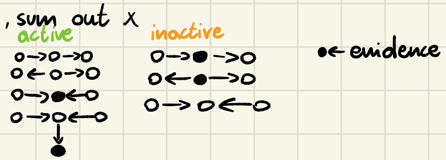
Hidden variables → not present in what we want to compute

representation → DAG A→B we store conditional table for P(B|A)
we only it for the parents

inference by enumeration → collect all rows consistent with evidence, sum out hidden variables
normalize
when calculating probabilities consider only directly connected nodes

better inference → eliminate variables one by one, to eliminate x
join (multiply) all factors involving X, sum out X

check indipendence → no active path ⇒ indipendence
otherwise no claim



sampling → repeated simulation, faster than inference
have random in [0,1], each outcome has interval assigned → probability of outcome

6) Decision networks

chance nodes -> identical to Bayes' nets. Outcome with prob.

Action nodes -> choice over actions, we have the power

utility node -> children of combinations of action and chance nodes, output utility based on the parents

goal -> select actions that lead to the maximum expected utility

- instantiate all evidence, run inference for posterior probability of all chance nodes parents to the utility node into which the action node feeds
- go through all possible actions and compute the expected utility

  $EU(a|e) = \sum_{x_1:x_n} P(x_1,\dots x_n|e)\, U(a, x_1,\dots,x_n)$
- select max

value of perfect information -> exptected increase in utility if new evidence observed

- non negative
- non additive
- order indipendent
- if $Parents(U) \perp\!\!\!\perp Z \mid current\ evidence$

  then $VPI(Z \mid current\ evidence) = 0$

7) Hidden Markov models

introduce time/space

state -> value of X at a given time

Parameters -> transition probabilities

conditional dependency -> past & future indipendent given present