Steps agent takes to solve the problem:

- **Goal Formulation** - Goals organize behavior by limiting the objectives and hence the actions to be considered.
- **Problem Formulation** - The agent devises a description of the states and actions necessary to reach the goal — an abstract model of the relevant part of the world
- **Search**: Before taking any action in the real world, the agent simulates sequences of actions in its model, searching until it finds a sequence of actions that reaches the goal. Such a sequence is called a *solution*. The agent might have to simulate multiple sequences that do not reach the goal, but eventually it will find a solution.
- **Execution**: The agent can now execute the actions in the solution, one at a time.

In a fully observable, deterministic, known environment, the solution to any problem is a fixed sequence of actions.

Search Problem A search problem consists of:

- state space: A set of possible states that the environment can be in
- initial state: The state in which the agent begins
- goal state: Sometimes there is one goal state , sometimes there is a small set of alternative goal states, and sometimes the goal is defined by a property that applies to many states. All three can be encompassed by an `IS-GOAL` state.
- actions: Given a state s, `ACTIONS`(s) returns a finite set of actions that can be executed in We say that each of these actions is applicable in s.
- transition model: A description of what each action does, specified by a function `RESULT`(s, a) that returns the state that results from doing action a in state s.
- action cost function: A function `ACTION-COST`(s, a, s') that assigns a numeric cost to each path, where s' is the state resulting from doing action a in state s.

A sequence of actions forms a **path**, and a **solution** is a path from the initial state to a goal state. An **optimal solution** has the lowest path cost among all solutions. The state space can be represented as a **graph** in which the vertices are states and the directed edges between them are actions.

Search Algorithms

- Node: A node in a search tree over the state-space graph. Initial state is root, Goal state is goal node.
- Expansion of a node: identify and generate child nodes
- Reaching a node: Once a node's existence is known (i.e., after parent's expansion)
- Evaluation function ($f(n)$): a function used to identify the next node to be selected and expanded.
- Frontier: Separates two regions of the state-space-graph - expanded nodes and nodes not yet reached (a priority queue, stack, queue, etc.)

Performance Mesasurement of search methods.

- Completeness: Is the algo guaranteed to find a solution when there is one, and correctly report failure when there isn't?
- Cost optimality: Does it find a solution with the lowest path cost of all solutions?
- Time Complexity: How long does it take to find a solution? This can be measured in seconds, or more abstractly by the number of states and actions considered.
- Space Complexity: How much memory is needed to perform the search?
- Implicit vs Explicit State-space:
  - Explicit state-space graph: complexity in |V| & |E|
  - Implicit state-space graph: complexity using Depth (or number of actions), number of maximum actions in any path or branching factor

Uninformer Search Algos

- BFS, DFS, UCS or Dijkstra's
- Iterative Deepening Search (IDS) - performs DFS with increasing depth limit until a solution is found (also called Depth limited search or DLS). It combines the space efficiency of DFS with the completeness of BFS.
- Bidirectional Search - simultaneously searches forward from the initial state and backward from the goal state, hoping that the two searches meet in the middle. It is complete and optimal if the search is done using breadth-first search.

b is the branching factor; m is the maximum depth of the search tree; d is the depth of the shallowest solution, or is m when there is no solution; l is the depth limit.

[1] complete if is finite, and the state space either has a solution or is finite. [2] complete if all action costs are $\geq \epsilon > 0$, [3] cost-optimal if action costs are all identical; [4] if both directions are breadth-first or uniform-cost.

Criterion	BFS	UCS	DFS	DLS	IDS	Bi
Complete?	Yes[1]	Yes[1,2]	No	No	Yes[1]	Yes[1,4]
Optimal Cost?	Yes[3]	Yes	No	No	Yes[3]	Yes[3,4]
Time	$\mathbb{O}(b^d)$	$\mathbb{O}(b^{1+C^*/\epsilon})$	$\mathbb{O}(b^m)$	$\mathbb{O}(b^l)$	$\mathbb{O}(b^d)$	$\mathbb{O}(b^{d/2})$
Space	$\mathbb{O}(b^d)$	$\mathbb{O}(b^{1+C^*/\epsilon})$	$\mathbb{O}(bm)$	$\mathbb{O}(bl)$	$\mathbb{O}(bd)$	$\mathbb{O}(b^{d/2})$

Informed Search Algos

- Greedy Best First Search - $f(n) = h(n)$. Complete in finite state spaces, but not optimal.
- A* search - $f(n) = g(n) + h(n)$, where $g(n)$ is path cost and $h(n)$ is heuristic cost. Pick node with minimal value of f to be expanded.

Completeness & Optimality of A* Search

- Admissible heuristic: one that never overestimates the cost to goal
- Consistent heuristic: $\forall n$ and every successor n' of n generated by an action a we have $h(n) \leq c(n, a, n') + h(n')$.
- Admissible heuristics are guaranteed to return an optimal path
- Inadmissible heuristics *may* return optimal paths if the following conditions are satisfied
  - If there is even one cost-optimal path on which $h(n)$ is admissible for all nodes n on the path, then that path will be found, no matter what the heuristic says for states off the path
  - If the optimal solution has cost $C*$, and the second-best has cost C_2 , and if $h(n)$ overestimates some costs, but never by more than $C * -C_2$, then A* is guaranteed to return optimal cost solutions.

P&S Stuff

$$F(x) = P(X < x) = \int_{-\infty}^{x} f(x)dx, \int_{-\infty}^{\infty} f(x)dx = 1$$

$$E(X) = \sum_x xf(x), E(X^2) = \sum_x x^2 f(x)$$

$$E(X) = \int_{-\infty}^{\infty} xf(x)dx, E(X^2) = \int_{-\infty}^{\infty} x^2 f(x)dx, Var(X) = E(X^2) - E(X)^2$$

$$E(Y|A) = \sum_y P(Y = y|A), E(Y|A) = \int_{-\infty}^{\infty} yf(y|A)dy$$

$$P(Y = y|X = x) = \frac{P(X = x, Y = y)}{P(X = x)}, f_{Y|X}(y|x) = \frac{f_{X,Y}(x, y)}{f_X(x)}$$

$$f_X(x|A) = \frac{P(A|X = x)f_X(x)}{P(A)}$$

$$P(X = x) = \sum_y P(X = x, Y = y), f_X(x) = \int_{-\infty}^{\infty} f_{X,Y}(x, y)dy$$

$$\phi(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Distrubution	CDF	PMF	$\mathbb{E}[X]$
Bernoulli	$(1-p)^{1-x}$	$p^x(1-p)^{1-x}$	p
Binomial	$\sum_{i=0}^{x} \binom{n}{i} p^i(1-p)^{n-i}$	$\binom{n}{x} p^x(1-p)^{n-x}$	np
Geometric	$1-(1-p)^x$	$p(1-p)^{x-1}$	$\frac{1}{p}$
Poisson	$e^{-\lambda} \sum_{i=0}^{x} \frac{\lambda^i}{i!}$	$\frac{\lambda^x e^{-\lambda}}{x!}$	λ