Admissible: n is a node, h is a heuristic
$h(n)$ is the cost indicated by h to reach a goal from n
$h^*(n)$ is the optimal cost to reach a goal from n.
$h(n)$ is admissible if $\forall n, h(n) \leq h^*(n)$
Consistent $\Rightarrow$ Admissible (not the other way around)

Consistent: if the estimate of a heuristic is always less than or equal to the estimated distance from any neighbouring vertex to the goal, plus the step cost of reaching that neighbour.

$$h(N) \leq c(N, P) + h(P) \quad \& \quad h(G) = 0$$

minmax
α-β pruning

Agents interact with environments through actuators & sensors
Agent function describes ~~the~~ what the agent does in all circumstances
Agent programs implement agent functions.
The performance measure ~~is~~ evaluates the environment sequence.
A perfect rational agent maximizes expected performance

PEAS descriptions define task environments
Environments are categorized along several dimensions:
observable? deterministic? known? episodic? static? discrete? single-agent?
Several basic agent architecture exist:
reflex, reflex with state, goal-based, utility-based → utility function internalizes the performance measure
current percept only (internal state keeps track of ...) → (planning & search, the agent ~~is~~ combines goal & env model to choose actions)
All agents can improve their performance through learning.

Summary of algorithms (uninformed)

expand least-cost unexpanded node

Criterion	BFS	Uniform Cost	DFS	Depth-Limited	IDS
Complete?	Yes*	Yes*	No	Yes, if $l \geq d$	Yes*
Time	b^{d+1}	$b^{1+\lceil C^*/\varepsilon \rceil}$	b^m	b^l	b^d
Space	b^{d+1}	$b^{1+\lceil C^*/\varepsilon \rceil}$	bm	bl	bd
Optimal?	Yes*	Yes	No	No	Yes*

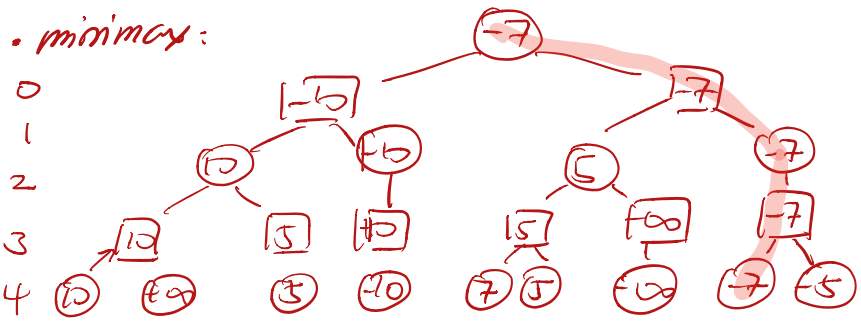
a problem { initial state; successor function $S(x)$ = set of action-state pairs; goal test; path cost }

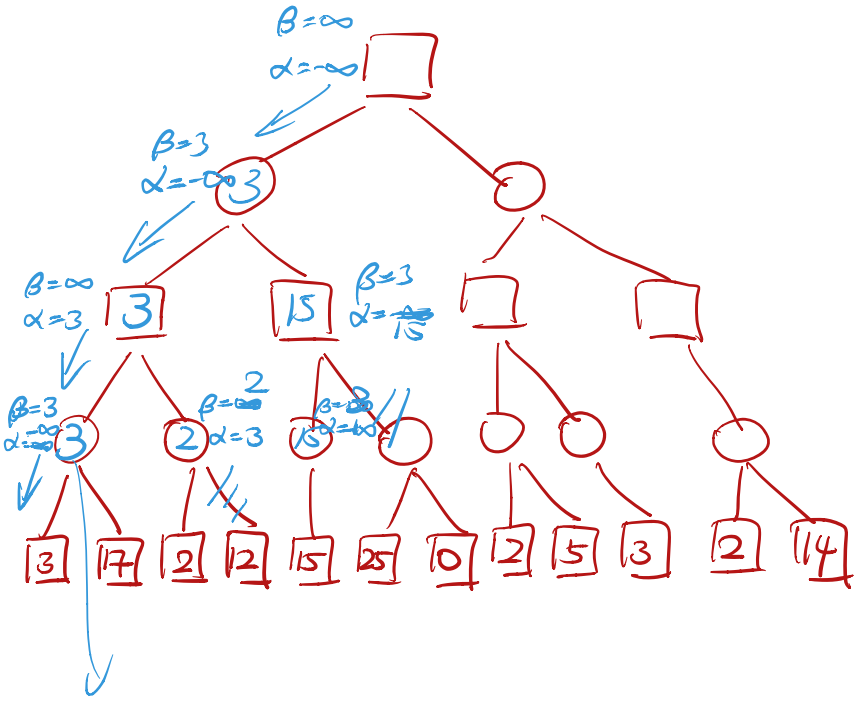
- Which algorithm to use?

strategy	solution	useful when	frontier	algorithm & state space
DFS	arbitrary	many solutions exist	LIFO	tree-search for finite acyclic graphs recursive algorithm for finite acyclic graphs add cycle detection for finite graphs tree search graph search (simple) may improve performance tree search for trees graph-search (simple) for equal step costs graph-search (optimal) for arbitrary step costs (no reopening needed)
BFS	shortest	shallow solution exist	FIFO	
UC	optimal	good admissible heuristics lacking	priority queue ordered by g	
A*	optimal	good admissible heuristics exist	ordered by f	
greedy search	arbitrary	good (inadmissible) heuristics exist	ordered by h	

- minimax: select optimal actions for two-player zero-sum games of perfect information by a DFS of the game-tree.
- alpha-beta pruning does not compromise optimality but increases efficiency by eliminating probably irrelevant subtrees
- cut the search off using evaluation function.

- minimax:





MAX

MIN

MAX

MIN

MAX