

World = environment + actions

reflex agent: selects an action based solely on the current state of the world.
 < planning agent: maintain a model of the world to simulate actions.

	Complete?	Optimal?	Time?	Space?
DFS	NO	NO	$O(b^m)$	$O(bm)$
BFS	YES	YES if UCS NO otherwise	$O(b^s)$	$O(b^s)$
UCS	YES	YES	$O(b^{c^*/\epsilon})$	$O(b^{c^*/\epsilon})$
Greedy	NO	NO	poorly guided DFS	
A*	YES	YES	BFS w/ bad heuristic	

Tree search + closed set = Graph search
 A* optimal w/ admissible h only set, not a list
 A* optimal & complete w/ consistent h only

- A* tree search w/ admissible heuristics will yield an optimal solution.
- A* graph search w/ only consistent heuristics will yield an optimal solution.

SEARCH STATE

- start state
- successor function
- goal state / test
- state space

path/ plan: path from start state to goal state
 strategy: order in which states are considered.

+ state space

graph: each state is represented exactly once

+ search tree: each node is a path to current state

b = branching factor m = max depth
 s = depth of shallowest goal node.
 C* = optimal path cost
 E = minimal cost b/w two nodes

HEURISTICS - admissibility, consistency

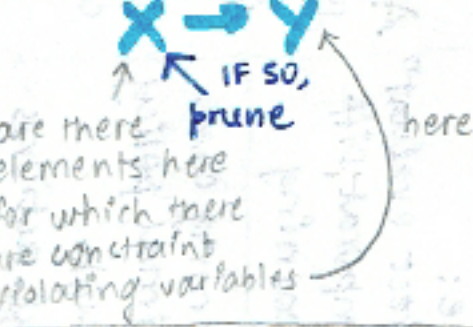
Admissibility: $\forall n, 0 \leq h(n) \leq h^*(n)$
 Consistency: $\forall A, C, h(A) - h(C) \leq \text{cost}(A, C)$
 $f(C) \geq f(A) + A^*gs$

- values of f(n) (g(n)+h(n)) for nodes along any plan are non decreasing.
- Dominance: max function applied to multiple admissible heuristics \rightarrow also admissible

CSP \rightarrow identification problem, no regard to how we arrive at goal state. NP-HARD
 N variables, domain size O(d) for each \Rightarrow total possible assignments $O(d^N)$

DFS < Backtracking search < Backtracking search w/ filtering < BTS + f w/ ordering
 1. fix an order for variables
 2. start assigning w/o conflict
 3. if no value, return & repeat

ARC CONSISTENCY



when to use	how to use	result
during the search, after assignment	unassigned neighbouring node	arc consistency only for the two nodes
during or before the search	enqueue & propagate then enqueue those that point to X.	full arc consistency

Minimum remaining values: Next unassigned variable has the fewest valid remaining values
 Least constraining value: Choose that value that prunes the fewest from the remaining.

TREE STRUCTURED CSP

- no loops in its constraint graph
 - reduce runtime $O(d^N) \rightarrow O(nd^2)$
- LINEAR ORDERING STEPS**
- Pick an arbitrary node as root
 - Order the nodes so that the edges point right from root (DAG)
 - BACKWARD PASS of arc consistency
 - FORWARD assignment of values.

A graph that is strong k-consistent possesses the property that any subset of k nodes is k, k-1, k-2, ... 1 consistent too.

CUTSET - assign to cutset, prune neighbours, solve. $O(d^{(n-c)}d^2)$

ITERATIVE IMPROVEMENT while NOT SOLVED: 1. randomly choose conflicted variable 2. choose a value that violates fewest constraints

Critical ratio = $\frac{\# \text{ constraints}}{\# \text{ variables}}$ for which II is expensive of.

MINIMAX (deterministic zero sum games) \sim DFS traversal

$V(s) = \max V(s')$ or known for non-terminal states for terminal } best possible outcome from here on
 $= \min V(s')$ \rightarrow if opponent controlled state

α - β PRUNING $O(b^{m/2})$
 MAX node: prune if $v > \beta$ } all finite
 MIN node: prune if $v < \alpha$ }
 α, β : best explored path from current node to root for MAXIMIZER, MINIMIZER

EVAL FUNCTIONS

Eval(s) = $\sum_i w_i f_i(s)$

$V(s) = \sum_{s'} p(s') V(s')$
 for all chance states
EXPECTIMAX
 Can never prune a chance node
 final utility no longer represents a node on a descendant

NON ZERO SUM GAMES

- nodes are tuples for various players
 - Co-op, compete thru computation