Contents

Chapter 1

Introduction

An **agent** is an entity that *perceives* and *acts*. An agent is a function from percept histories to actions:

$$f : \mathcal{P}^* \to \mathcal{A} \tag{1.1}$$

A rational agent chooses whichever action maximizes the expected value of performance measure given the precept sequence to date.

Types of agents

Reflex agent choose action based on current percept (and maybe memory). Act oh how the world is.

Goal-based agent decision based of hypothesized consequence of action. Act on how the world would be.

Utility-based agent Trade off multiple goals. Act on how the world will likely be.

1.1 Uninformed search

In Tree seach algorithms, every node responds to a state and the path to get to that node from the root, plan. A strategy is defined by picking the order of node expansion. A strategy is evaluated by following factors

Completeness always finds a solution.

Time complexity number of nodes expanded

Space complexity maximum number of nodes in memory

Optimality finds the best solution

Time and space complexity are measured in terms of

b maximum branching factor of search tree

d depth of the least-cost solution.

m maximum depth of the state space

Uninformed startegies use only the information available in the problem definition. In tree search we expend current state and generate new set of states. In graph search visited nodes are not added to frontier, instead added to explored sets.

Breadth-first search expand the shallowest unexpanded node

- **Complete** yes (if b, d are finite).
- **Time** $O\left(b^{d+1}\right)$.
- **Space** $O\left(b^{d+1}\right)$, keeps every node in memory.
- **Optimal** Yes (if cost = 1 per step or non-decreasing function of depth), not optimal in general

Uninform-cost search Cost function is $\sum_{i=root}^{node} c_i$

- **Complete** yes (if step cost $\geq \epsilon$).
- **Time** $O\left(b^{1+\lfloor C^*/\epsilon \rfloor}\right)$, C^* is the cost of the optimal solution. This can be worse than BFS.
- **Space** $O\left(b^{1+\lfloor C^*/\epsilon \rfloor}\right)$.
- **Optimal** Yes- nodes expanded in increasing order of cost.

Depth-first search expand the deepest unexpanded node.

- **Complete** no fails in infinite-depth spaces, or spaces with loops.
- **Time** $O(b^m)$.
- **Space** $O(bm)$ and by backtracking it can be reduced to $O(m)$.
- **Optimal** No.

Depth-limited search is combination of BFS and DFS. A DFS with a depth limit l.

- **Complete** yes if $d \leq l$
- **Time** $O\left(b^l\right)$
- **Space** $O(bl)$ and by backtracking it can be reduced to $O(l)$.
- **Optimal** probably not.

Iterative deepening search gradually increasing the depth limit in DLS.

- **Complete** Like BFS if b, d are finite. no fails in infinite-depth spaces, or spaces with loops.
- **Time** $O\left(b^d\right)$.
- **Space** $O(bd)$.
- **Optimal** No (like BFS the cost function needs to be non-decreasing), but it can be modified to explore uniform-cost (Iterative lengthening).

In graph search we visit each state once, however the closed list space can become exponential.

1.2 Informed search

Also, a **heuristic** is given to us. A heuristic is an estimate of how close a state is to a goal.

1.2.1 best-first search, greedy

Cost function is applied to each node. Heuristic function, estimated cost of the cheapest path from state at node n to a goal state. It is a component of f. $h(n) = f(n)$

Complete Not really, might get stuck in a dead end

Time $O(b^m)$.

Space $O(b^m)$.

Optimal Not really.

1.2.2 A^* search

A best-first seach with $f(n) = g(n) + h(n)$ where $g(n)$ is the sum of costs from start to n and $h(n)$ is the estimate of lowest cost path n, the heuristic.

Complete If h satisfies the following condition.

Optimal If h satisfies the following condition.

1. $h(n)$ is admissible: never over-estimates the cost to get to the goal. Hence $f(n)$ never overestimates as g is the actual cost.
2. $h(n)$ must be consistent, that is for any node
$$h(n) \leq c(n, a, n') + h(n')$$
for all actions a from node n to any of its successor n'.

Proposition 1.1. *consistency* $\implies$ *admissibility.*

Optimality of A^*

On tree $h(n)$ must be admissible and on graph $h(n)$ it must be consistent. To prove the second proposition consider the followings

- $f(n)$ is increasing on any graph
$$f(n') = h(n') + g(n') = h(n') + c(n, a, n') + g(n) \geq h(n) + g(n) = f(n)$$
- Whenever A^* selects a node for expansion, the optimal path for that node is found; if not true the must have been a cheaper path to n and thus there need to be n' in the frontier on another path from root to n. But by the lemmes $f(n') \leq f(n)$ so it must have been expanded before n. In fact, A^* expands all nodes with $f(n) < C^*$ and might expand some of the nodes with $f(n) = C^*$.

Completeness of A^*

It is equivalent to the finiteness of $\{n \,|\, f(n) \leq C^*\}$ which means step costs are greater than some ϵ and b is finite.

1.3 Memory-bounded heuristic search

Iterative deepening A^* cutoff for $f(n)$ instead of depth.

Recursive best-first search limited DFS (Recursive).

MA* caches the nodes and deletes the “worst” when it is full.