Lecture 5: Informed Search

Adam Hawley

January 28, 2019

1 Informed vs. Uninformed Search

Search algorithms differ by how they pick which nodes to expand and when.

1.1 Uninformed Search Algorithms

In making this decision, these look only at the structure of the search tree and not at the statess inside the nodes These are often also called **blind** searcj algorithms. We have seen 5 of these in lecture 3.

1.2 Informed Search Algorithms

In making this decision these look at the states inside the nodes. Also sometimes called **heuristic** search algorithms. These will be covered in this lecture.

2 Informed Search Algorithms in Detail

2.1 Algorithm i: Best-First Search

Main idea: Use an *evaluation function* $f(n)$ for each node n. This function estimates the *desirability* of expanding that node. And then you expand the *most desirable* unexpanded node.

Implementation: Order the nodes in the fringe L in decreasing order of desirability. Then always choose nodes from the front of L.

The evaluation $f(n)$ will be $h(n)$, the function which gives the value of the heurisitc e.g the straight line distance. Greedy best-first search expands the node that appears to be closest to the goal.

- Completeness: There are a variety of cases where completeness is lost e.g looping.
- Time complexity: $O(b^m)$, but a good heuristic can give dramatic improvement
- Space comlpexity: $O(b^m)$ — keeps all nodes in memory
- Optimal: No since sometimes it doesn't find a solution at all.

2.2 A* Search

Main idea: avoid expanding paths that are already expensive. Here the evaluation function $f(n) = g(n) + h(n)$ where

- $g(n)$ is the cost so far to reach n.
- $h(n)$ is the estimated cost from n to the nearest goal state.
- $f(n)$ is the estimated total cost of path through n to a goal state.

Note: $f(n) = g(n)$ is used by uniform cost search and $f(n) = h(n)$ is used by greedy best-first search.

3 Admissable Heuristics

A heuristic is admissable if for every node n, $h(n) \leq h^*(n)$, where $h^*(n)$ is the true cost to reach the nearest goal state from n. Therefore, an admissable heuristc nevver overestimates the cost to reach the goal i.e it is optimistic.

3.1 Dominance of Heuristics

If $h_2(n) \geq h_1(n)$ for all n and both $h1$ and $h2$ are admissable, then h_2 is said to "dominate" h_1.

Theorem: If h_2 dominates h_1 then A* with h_1 expands every node expanded by A* with h_2 and possibly more.

3.2 How to Obtain Admissable Heuristics?

A problem with fewer restrictions on the actions is called a *relaxed problem.* The cost of an optimal solution to a relaxed problem is an admissable heuristic for the orignal problem.

3.3 Consistent Heuristics

A heuristic is consistent if for every node n, every successor n' of n generated by any action a, $h(n) \leq c(n, a, n') + h(n')$.

If h is consistent, we have:

$$\begin{aligned} f(n') &= g(n') + h(n') \\ &= g(n) + c(n, a, n') + h(n') \\ &\geq g(n) + h(n) \\ &\geq f(n) \end{aligned}$$

i.e, $f(n)$ is non-decreasing along any path, that is, $f(n)$ is **monotonic**.

Theorem: If a heuristic is consistent then it is admissable.

Theorem: If $h(n)$ is consistent, A* can be implemented more efficiently — no need ever to reevaluate the cost of already expanded nodes).

4 Tree Search vs. Graph Search

- Graph search solves the problem with revisting states
- Efficient probkem representation & tree search may be enough.
- DFS/IDS using tree search could check the current path to eliminate looops
- Graph search stores nodes corresponding to all visited states.