Artificial Intelligence for Robotics - Assignment 04

Deepan Chakravarthi Padmanabhan
Muhammad Umer Ahmed Khan

November 2018

Artificial Intelligence-04 assignment

1. Give an explanation to prove the following statements:

   - **Breadth-first search is a special case of uniform-cost search:**
     Uniform cost search(UCS) utilizes a priority queue managed depending on the cumulative cost to reach the node. Uniform Cost Search gives the minimum cumulative cost the maximum priority. In Uniform cost search, the evaluation function that guides the search f(n) is given as below:
     Let the path be considered as: 1-2-3.

     $$f(n) = g(n) = \text{path cost of node 1-2} + \text{path cost of node 1-2-3}$$

     Here, g(n) is the total cost to reach a node 'n' from the start and no heuristic function is used here. UCS expands the node with the lowest g(n).
     BFS primarily visits the node with the shortest path length (number of nodes) from the root node, UCS first visits the node with the shortest path costs (sum of edge weights) from the root node. Here, BFS explores vertices in the order of their distance from the source vertex. The distance is the minimum length of a path from source vertex to the node.
     Breadth First Search(BFS) is a special case of uniform-cost search when all edge costs are positive and identical. In this case, the evaluation function is given as below: For instance, step cost =1

     $$f(n) = g(n) = 1 * depth(n)$$

   - **Breadth-first search, depth-first search, and uniform-cost search are special cases of Greedy Best-First Search:**
     BFS, DFS and UCS are uninformed search methods. However, all the searches are guided by the evaluation function f(n). The uninformed search methods- BFS, DFS and UCS are special cases of Greedy Best-First search. The f(n) of Greedy Best-First Search is given by:

     $$f(n) = h(n)$$

     h(n) is the heuristic function. This is an estimation of cost from n state to the closest goal. The below f(n) illustrates the BFS, DFS and UCS working conditions of Greedy Best first search .

     f(n)= depth(n) ; then Greedy Best First Search works as BFS
     f(n)= -depth(n) ; then Greedy Best First Search works as DFS
     f(n)= g(n) ; then Greedy Best First Search works as UCS

   - **Uniform-cost search is a special case of A* search:**
     The evalution function for A* search is given as:

     $$f(n) = g(n) + h(n)$$

     Here g(n) is the actual cost to reach node n from the start and h(n) is the estimate of cost from n to the closest goal. A* Search reduces to uniform cost search when the heuristic function is zero everywhere, i.e. h(n) = 0 for all n. This heuristic is clearly admissible since it always underestimates the distance remaining to reach the goal. Thus, UCS is A* with h(n)=0.

2. Answer the following questions regarding A* search:

   - When is A* complete?
     The 2 completeness conditions of A* states are- no infinite paths have finite cost and consistency of the heuristic. When the heuristic is inconsistent, the graph-search algorithm has to be modified to ensure completeness. The consistency of a heuristic is analogous to the triangular inequality and ensures f(n) increases along any path. Hence, if there are no infinitely many nodes with f $\leq$ f (G), where G indicates the shortest optimal goal, the algorithm is complete. This ensures that A* returns a solution.
   - When does A* end the search process?
     A* determines the expansion path based on the the cost of the path and an estimate of the cost required to extend the path all the way to the goal. Specifically, A* selects the path that minimizes

     $$f(n) = g(n) + h(n)$$

where n is the next node on the path, g(n) is the cost of the path from the start node to n. h(n) is a heuristic function that estimates the cost of the cheapest path from n to the goal. A* terminates when the path it chooses to extend is a path from start to goal or if there are no paths eligible to be extended.The heuristic function is problem-specific. If the heuristic function is admissible, h(n) never overestimates the actual cost to get to the goal, A* is guaranteed to return a least-cost path from start to goal.
A* algorithm expands all nodes with f (n) $<$ C*, where C* is the cost of the optimal solution path. Sometimes, it might then expand some of the nodes right on the goal contour (where f (n) = C*) before selecting a goal node. If this is satisfied, the algorithm ends the searching process. A* expansion criteria is as given below:

If f(n) is consistent, A* does not expand nodes for f(n) $>$ C*
A* expand nodes for f(n) $<$ C*
A* expand some nodes for f(n) = C*