

Heuristic Search Study Guide

Artificial Intelligence

1 Heuristic Search

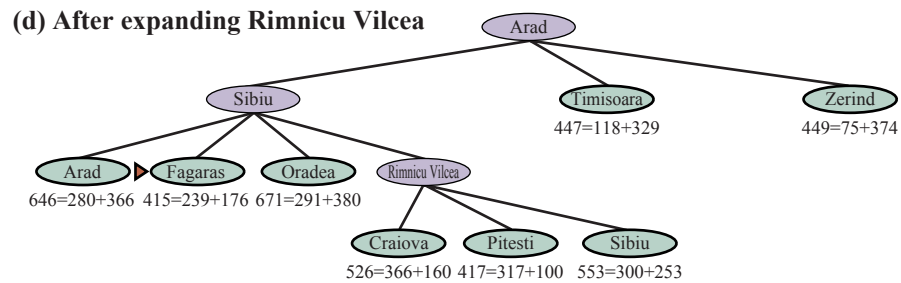
1. What is a heuristic function?

Solution: A heuristic function, $h(n)$, gives an estimated cost of the cheapest path from the state at node n to a goal state.

2. How do you use Best-First Search to implement A^* search?

Solution: Remember that Best-First Search uses a priority queue to choose the next node for consideration, where the ordering of nodes in the priority queue is determined by the value of a function $f(n)$. If the path cost from the initial state to a node n is given by $g(n)$, then $f(n) = g(n) + h(n)$ is an estimate of the complete path cost from the initial state to a goal state through node n . A^* search uses $f(n) = g(n) + h(n)$ to order the nodes in the priority queue.

3. In this diagram, nodes are marked with their A^* heuristic values, $f(n) = g(n) + h(n)$. Which node will be the next node expanded by A^* ?



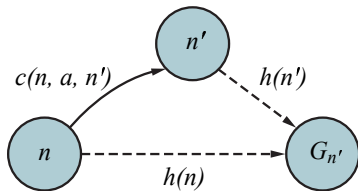
Solution: Fagaras, because it has the least $f(n)$ value of all the unexpanded nodes.

4. Define admissible heuristic.

Solution: An admissible heuristic never overestimates the cost of the path from a node a goal

5. Define consistent heuristic.

Solution: A heuristic $h(n)$ is consistent if, for every node n and every successor n' of n generated by an action a , we have $h(n) \leq c(n, a, n') + h(n')$.



6. Which important property does an admissible heuristic give A^* ?

Solution: With an admissible heuristic, A^* is cost-optimal.

7. How does using a consistent heuristic improve A^* compared to using an admissible, but not consistent heuristic?

Solution: With a consistent heuristic, the first time A^* reaches a state it will be on an optimal path, so it never re-adds a state to the frontier, never changes an entry in **reached**.

8. Describe three approaches to designing a heuristic function.

Solution:

- Relaxing the problem definition
- Storing precomputed solution costs for subproblems in a pattern database
- Defining landmarks
- Learning from experience

9. Use one of the three approaches from the previous question to design an admissible heuristic for the Three Pitchers problem. In the Three Pitchers problem there are three unmarked pitchers – an 8 L pitcher full of water, an empty 5 L pitcher, and an empty 3 L pitcher – and an agent must reallocate the water so that one of the pitchers contains exactly 4 L of water.

Solution: