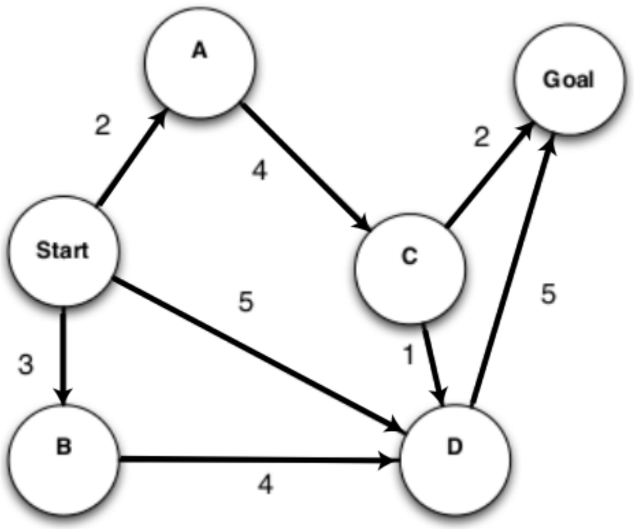
CS461 Artificial Intelligence

Fall 2024: Homework 1

Due Date: 20.10.2024

1 Uninformed Search



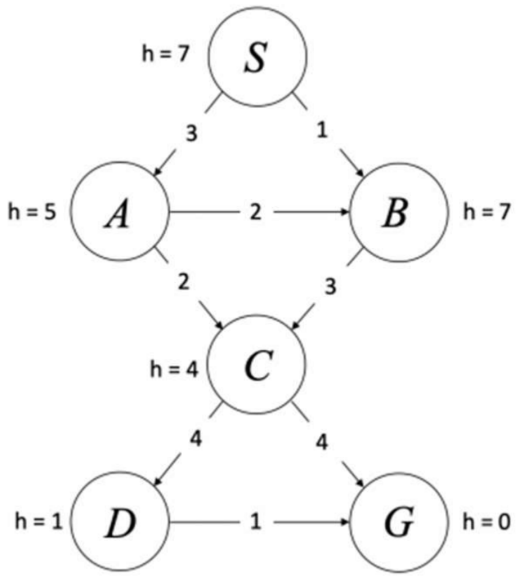
For each of the following graph search strategies, work out the order in which states are expanded, as well as the path returned by graph search. In all cases, assume ties resolve in such a way that states with earlier alphabetical order are expanded first. Remember that in graph search, a state is expanded only once.

(a) Depth-first search.

(b) Breadth-first search.

(c) Uniform cost search.

2 Informed Search



We will investigate various informed search algorithms for the following graph. Edges are labeled with their costs, and heuristic values h for states are labeled next to the states. S is the start state, and G is the goal state. In all search algorithms, assume ties are broken in alphabetical order.

(a) Select all boxes that describe the given heuristic values. If you think they are not admissible and/or consistent, give a counterexample to show it.

☐ admissible ☐ consistent ☐ neither

(b) Given the above heuristics, what is the order that the states are going to be expanded in, assuming we run greedy graph search with the heuristic values provided.

Index	1	2	3	4	5	Not Expanded
S	❍	❍	❍	❍	❍	❍
A	❍	❍	❍	❍	❍	❍
B	❍	❍	❍	❍	❍	❍
C	❍	❍	❍	❍	❍	❍
D	❍	❍	❍	❍	❍	❍
G	❍	❍	❍	❍	❍	❍

(c) Assuming we run greedy graph search with the heuristic values provided, what path is returned?

(d) Given the above heuristics, what is the order that the states are going to be expanded in, assuming we run A* graph search with the heuristic values provided.

Index	1	2	3	4	5	Not Expanded
S	❍	❍	❍	❍	❍	❍
A	❍	❍	❍	❍	❍	❍
B	❍	❍	❍	❍	❍	❍
C	❍	❍	❍	❍	❍	❍
D	❍	❍	❍	❍	❍	❍
G	❍	❍	❍	❍	❍	❍

(e) Assuming we run A* graph search with the heuristic values provided, what path is returned?

3 Iterative Deepening A*

Consider a variant of iterative deepening called iterative deepening A*, where instead of limiting the depth first search by depth as in standard iterative deepening search, we can limit the depth-first search by the f **value as defined in A* search. As a reminder** $f[node] = g[node] + h[node]$ **where** $g[node]$ **is the cost of the path from the**

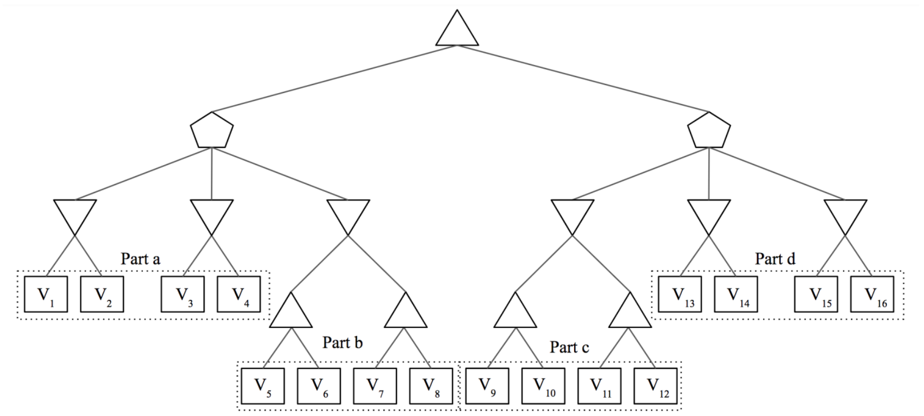
start state and $h[node]$ **is a heuristic value estimating the cost to the closest goal state.**

Assuming there are no ties in f **value between nodes, which of the following statements about the number of nodes that iterative deepening A* expands is True? If the same node is expanded multiple times, count all of the times that it is expanded. If none of the options are correct, mark None of the above. In any case, explain your reasoning below.**

- ❑ The number of times that iterative deepening A* expands a node is greater than or equal to the number of times A* will expand a node.
- ❑ The number of times that iterative deepening A* expands a node is less than or equal to the number of times A* will expand a node.
- ❑ We don't know if the number of times iterative deepening A* expands a node is more or less than the number of times A* will expand a node.
- ❑ None of the above

4 Adversarial Search

Consider a game tree with 3 kinds of nodes: maximizers (up triangles), minimizers (down triangles), and "medianizers" (pentagons). Medianizers select the median of their children. We say that a node is prunable if there is *any* possible configuration of utilities where that node's utility is guaranteed to not affect the root utility; the configuration for prunability does not have to be the same for two different nodes. Notice that alpha-beta pruning is not applicable here, however with a similar reasoning we can find conditions where a node can be pruned. You may assume that leaf nodes are evaluated left-to-right, and V_i's are unique.



(a) Determine which nodes labeled as ”Part a” are prunable, and explain your reasoning below.

Node	Prunable	Not Prunable
V_1	❍	❍
V_2	❍	❍
V_3	❍	❍
V_4	❍	❍

(b) Determine which nodes labeled as ”Part b” are prunable, and explain your reasoning below.

Node	Prunable	Not Prunable
V_5	❍	❍
V_6	❍	❍
V_7	❍	❍
V_8	❍	❍

(c) Determine which nodes labeled as ”Part c” are prunable, and explain your reasoning below.

Node	Prunable	Not Prunable
V_9	❍	❍
V_{10}	❍	❍
V_{11}	❍	❍
V_{12}	❍	❍

(d) Determine which nodes labeled as ”Part d” are prunable, and explain your reasoning below.

Node	Prunable	Not Prunable
V_{13}	❍	❍
V_{14}	❍	❍
V_{15}	❍	❍
V_{16}	❍	❍

5 Monte Carlo Tree Search

Answer the following questions about MCTS.

(a) Explain the key steps of the vanilla Monte Carlo Tree Search (MCTS) algorithm. How does each step contribute to the overall search process?

(b) Describe how the Upper Confidence Bound (UCB) formula is used during the selection phase of MCTS. What is the intuition behind balancing exploration and exploitation? What happens if you set the constant c too high or too low?

(c) How do progressive widening and double progressive widening work? In what types of problems would you expect progressive widening to be more effective than a standard MCTS approach? Provide an example problem.

(d) When are UCB and progressive widening useful? When can it be more beneficial to *not* use UCB and progressive widening?