DQ3 (L3)

Due No due date **Points** 25 **Questions** 9 **Time limit** None
Allowed attempts Unlimited

Instructions

- This quiz is NOT GRADED. However, it is HIGHLY RECOMMENDED that you use these questions to complement your review of the lecture content.
- The questions are based on content from the Lecture 3 and from part of Chapter 3 of the AIMA (4th Ed.) textbook (i.e., 3.5).

Take the quiz again

Attempt history

	Attempt	Time	Score
KEPT	Attempt 2	7 minutes	25 out of 25
LATEST	Attempt 2	7 minutes	25 out of 25
	Attempt 1	17 minutes	16.83 out of 25

Submitted 23 Jan at 17:29

Question 1

2 / 2 pts

Under the context of Informed Search, choose the correct option which best explains the following terms: g(n), h(n), f(n).

- **Specification A**
  - g(n): The function g(n) denotes the approximated path cost from an initial state, s, to state n. This path cost is calculated based on one specific path between s and n.
  - h(n): The function h(n), known as a heuristic, denotes the actual path cost from state n to (the closest) goal state g.
  - f(n): The function f(n) denotes the evaluation function that is adopted by the informed search algorithm in question - e.g., in the case of the greedy best-first search algorithm, f(n) = g(n) +

h(n).

- **Specification B**
  - g(n): The function g(n) denotes the actual path cost from an initial state, s, to state n. This path cost is calculated based on one specific path between s and n.
  - h(n): The function h(n), known as a heuristic, denotes the approximated path cost from state n to (the closest) goal state g.
  - f(n): The function f(n) denotes the evaluation function that is adopted by the informed search algorithm in question - e.g., in the case of the greedy best-first search algorithm, f(n) = g(n) + h(n).

- **Specification C**
  - g(n): The function g(n) denotes the actual path cost from an initial state, s, to state n. This path cost is calculated based on one specific path between s and n.
  - h(n): The function h(n), known as a heuristic, denotes the approximated path cost from state n to (the closest) goal state g.
  - f(n): The function f(n) denotes the evaluation function that is adopted by the informed search algorithm in question - e.g., in the case of the greedy best-first search algorithm, f(n) = h(n).

Specification A

Specification B

Correct!

Specification C

None of the above options.

Specification A is incorrect since:

- g(n) should correspond to actual and not approximated path cost from s to n
- h(n) should (generally) correspond to the approximated and not actual path cost from n to g
- greedy best-first search uses f(n) = h(n), not what is stated

Specification B is incorrect since:

- greedy best-first search uses f(n) = h(n), not what is stated

Specification C is correct since it does not contain the errors listed in Specifications A or B above

Question 2

3 / 3 pts

Which of the following is true about heuristics?

- [ ] If h1 is a consistent heuristic and h2 is an admissible heuristic, then the minimum of the two heuristics must be consistent.

Correct!

- [x] The minimum of an admissible heuristic and an inadmissible heuristic is admissible.

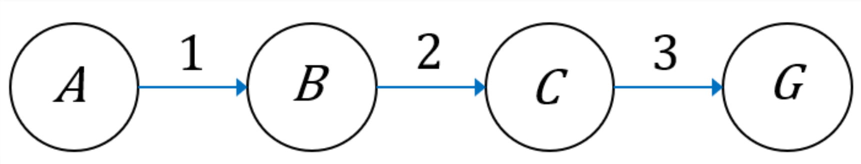
Correct!

- [x] A heuristic where at all nodes n, h(n) = 0, is considered an admissible heuristic.

- [ ] The minimum of 2 inadmissible heuristic is inadmissible.

- Recall that admissibility does not imply consistency. Taking the minimum of a consistent heuristic and an admissible heuristic will not lead to a consistent heuristic.

  Consider the following counter example.



	A	B	C	G	Notes
$h^*(n)$	6	5	3	0	$h^*(n)$ is the true optimal cost from n to its nearest goal
$h_1(n)$	6	5	3	0	consistent; for each $n \rightarrow_a n'$, $h_1(n) \le c(n, a, n') + h_1(n')$
$h_2(n)$	6	0	3	0	admissible; for each n, $h_2(n) \le h^*(n)$; this is not consistent since $h_2(A) > c(A, a, B) + h_2(B)$

- Only one h(n) value needs to overestimate the path cost to the nearest goal for h to be inadmissible. As such, if another inadmissible heuristic does not overestimate the path cost to the nearest goal at n, then the minimum of these two inadmissible heuristics would be admissible.

Question 3

4 / 4 pts

Consider two admissible heuristics h_1 and h_2, where h_2 dominates h_1.

Now consider the following heuristics:

$h_3 = (h_1 + h_2) / 2$

$h_4 = h_1 + h_2$

Determine the admissibility of h_3 and h_4.

Correct!

- [x] h_3 is admissible
- [ ] h_4 is admissible

- [ ] h_3 is inadmissible
- [ ] h_4 is inadmissible
- [ ] The admissibility of h_3 cannot be determined

Correct!

- [x] The admissibility of h_4 cannot be determined

h_3 will be admissible. The average of two admissible heuristics is admissible since it will never overestimate the cost of reaching the goal.

Dominance: $h_1 < h_3 < h_2$

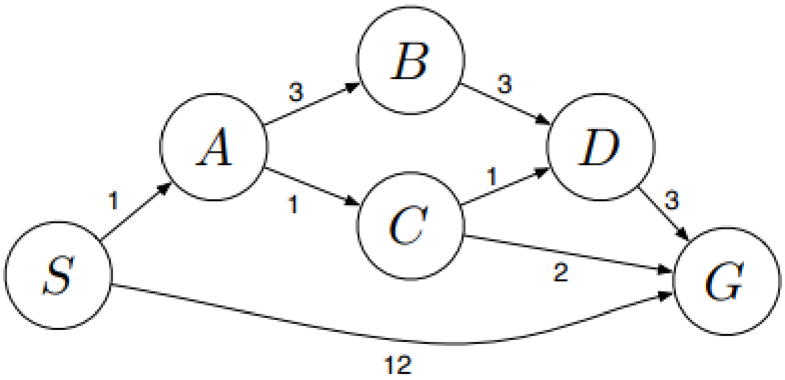
The admissibility of h_4 cannot be determined in this case. We are not given enough information to determine whether $h_1 + h_2$ will overestimate the cost of reaching the goal. For example, we can take $h_1(n) = 0$ for all n. In this case, $h_4(n) = h_1(n) + h_2(n) = h_2(n)$ for all n, since $h_1 = 0$. Thus, h_4 will be admissible in this example. However, we can simply find other examples to show that the sum of two admissible heuristics is inadmissible.

Question 4

4 / 4 pts

Consider the following scenario.

The diagram consists of a graph, with its nodes and step costs respectively. The table consists of two heuristics, h_1 and h_2, and their respective h(n) values at each node.



State	h_1	h_2
S	5	4
A	3	2
B	6	6
C	2	1
D	3	3
G	0	0

Which of the following is true?

- [ ] h_1 is admissible.

Correct!

- [x] h_2 is admissible.
- [ ] h_1 is consistent.
- [ ] h_2 is consistent.

h_1 is admissible: **False.** $h_1(S) = 5$ and $h^*_1(S) = 4$ (S -> A -> C -> G). Since the $h_1(S) > h^*_1(S)$, h_1 cannot be admissible.

h2 is admissible: **True.** Trace and compare the $h_2(n)$ values at each node with its respective $h^*_2(n)$ values. $h_2(n) <= h^*_2(n)$ for all n, hence h_2 is admissible.

h_1 is consistent: **False.** $h_1(S) = 5$, $d(S, A) = 1$ and $h_1(A) = 3$. Since $h_1(S) >= d(S, A) + h_1(A)$ (i.e., 5 >= 1 + 3), h_1 cannot be consistent.

h2 is consistent: **False.** $h_2(S) = 4$, $d(S, A) = 1$ and $h_2(A) = 2$. Since $h_1(S) >= d(S, A) + h_1(A)$ (4 >= 1 + 2), h2 cannot be consistent.

Question 5

2 / 2 pts

Which of the following statements about the Best-First Search Algorithm is true?

The Best-First Search algorithm is defined on page 91 of AIMA 4th Ed. It is as follows.

```
function BEST-FIRST-SEARCH(problem, f) returns a solution node or failure
  node ← NODE(STATE=problem.INITIAL)
  frontier ← a priority queue ordered by f, with node as an element
  reached ← a lookup table, with one entry with key problem.INITIAL and value node
  while not IS-EMPTY(frontier) do
    node ← POP(frontier)
    if problem.IS-GOAL(node.STATE) then return node
    for each child in EXPAND(problem, node) do
      s ← child.STATE
      if s is not in reached or child.PATH-COST < reached[s].PATH-COST then
        reached[s] ← child
        add child to frontier
  return failure

function EXPAND(problem, node) yields nodes
  s ← node.STATE
  for each action in problem.ACTIONS(s) do
    s′ ← problem.RESULT(s, action)
    cost ← node.PATH-COST + problem.ACTION-COST(s, action, s′)
    yield NODE(STATE=s′, PARENT=node, ACTION=action, PATH-COST=cost)
```

Correct!

☑ A* Search is a specific version of the Best-First Search Algorithm.

Best-First Search Algorithm uses a heuristic function h(n) to guide us to the goal.

Correct!

Best-First Search uses a priority queue that is ordered by non-decreasing cost of f(n).

Correct!

Uniform-Cost Search is a specific version of Best-First Search.

Option 2: It should be evaluation function f(n) instead of h(n). Only the greedy-best-first search algorithm uses f(n) = h(n).

All the other options are True.

Question 6

2 / 2 pts

Which of the following statements about Greedy Best-First Search is true?

Correct!

Greedy Best-First Search does not take into consideration the path cost that was taken to go to the node.

A graph search implementation of Greedy Best-First Search is optimal on state spaces that are finite.

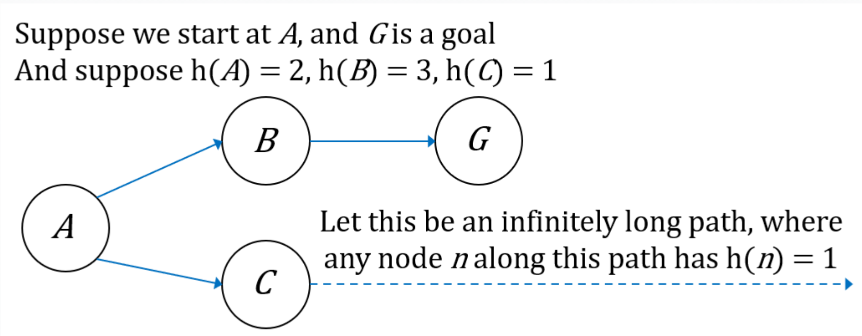
A graph search implementation of Greedy Best-First Search is always complete on state spaces where a solution exists.

Greedy Best-First Search has a space complexity of O(bd), where d is the depth of the shallowest goal node.

1. Recall that for the Greedy Best First Search algorithm, f(n) = h(n). This implies that path costs are ignored in Greedy Best-First Search.

2. As the Greedy Best First Search algorithm ignores path costs, it will not be optimal (refer to Tutorial 2 question 1c).

3. Even with a graph search implementation on a problem where a solution exists (i.e., there is a reachable goal), the greedy-best-first search may never reach this if the state space is infinite. Consider the following example.

   In this example, greedy-best-first search would simply traverse down the infinite path and never get to B.

4. The worst-case time and space complexity is $O(b^m)$ where m is the **max depth and not the shallowest depth (d)**.

Question 7

2 / 2 pts

Which of the following statements regarding the A* Search Algorithm are true?

You should assume that all action costs are > ε > 0, and the search space either has a solution or is finite.

A* Search is always complete.

A* Search is always optimal.

Correct!

When A* Search selects a node n for expansion, the shortest path to n may not been found yet.

Correct!

An admissible but inconsistent heuristic cannot guarantee optimality of A* using graph search Version 3.

Option 1 is False. Notice that in the given assumption state that the search space MAY be finite (if there is no solution), which implies that the branching factor, b, and finite maximum depth, m, MAY be finite. Suppose then that a solution exists but b is infinite. A* will not be complete in this case.

Also note that in several previous iterations, the term **state space** was instead used. However, we have omitted this term since it adds unwarranted complexity in understanding search algorithm properties. That said, you should consider what would happen if b and/or m is infinite given that all action costs are > ε > 0, and either graph or tree search is implemented.

The Options 2 and 3 are false and true respectively based on known properties of A* search.

The last option is true because

1. **Graph search (version 3) discards new paths to a repeated state. It may thus discard the optimal path.**
2. **A consistent heuristic ensures that f costs to be monotonically increasing along a path.**

Question 8

3 / 3 pts

Determine if the following statement is true or false

Depth-First Search will always expand at least as many nodes as A* Search with an admissible heuristic.

True

Correct!

False

False. Depth-first search may expand fewer nodes than A* search with an admissible heuristic.

E.g., Depth-first search may trace directly to the goal regardless of the value of an evaluation function.

Question 9

3 / 3 pts

Determine if the statement above is True or False.

Suppose that the A$*$ search algorithm utilises $f(n) = w \times g(n) + (1 - w) \times h(n)$, where $0 \leq w \leq 1$ (instead of $f(n) = g(n) + h(n)$). For any value of w, an optimal solution will be found whenever h is a consistent heuristic.

Assume that either a tree search or graph search (Version 3) implementation may be adopted.

True

Correct!

False

This is **false**. When w = 0 we just get greedy search, which is not guaranteed to be optimal.