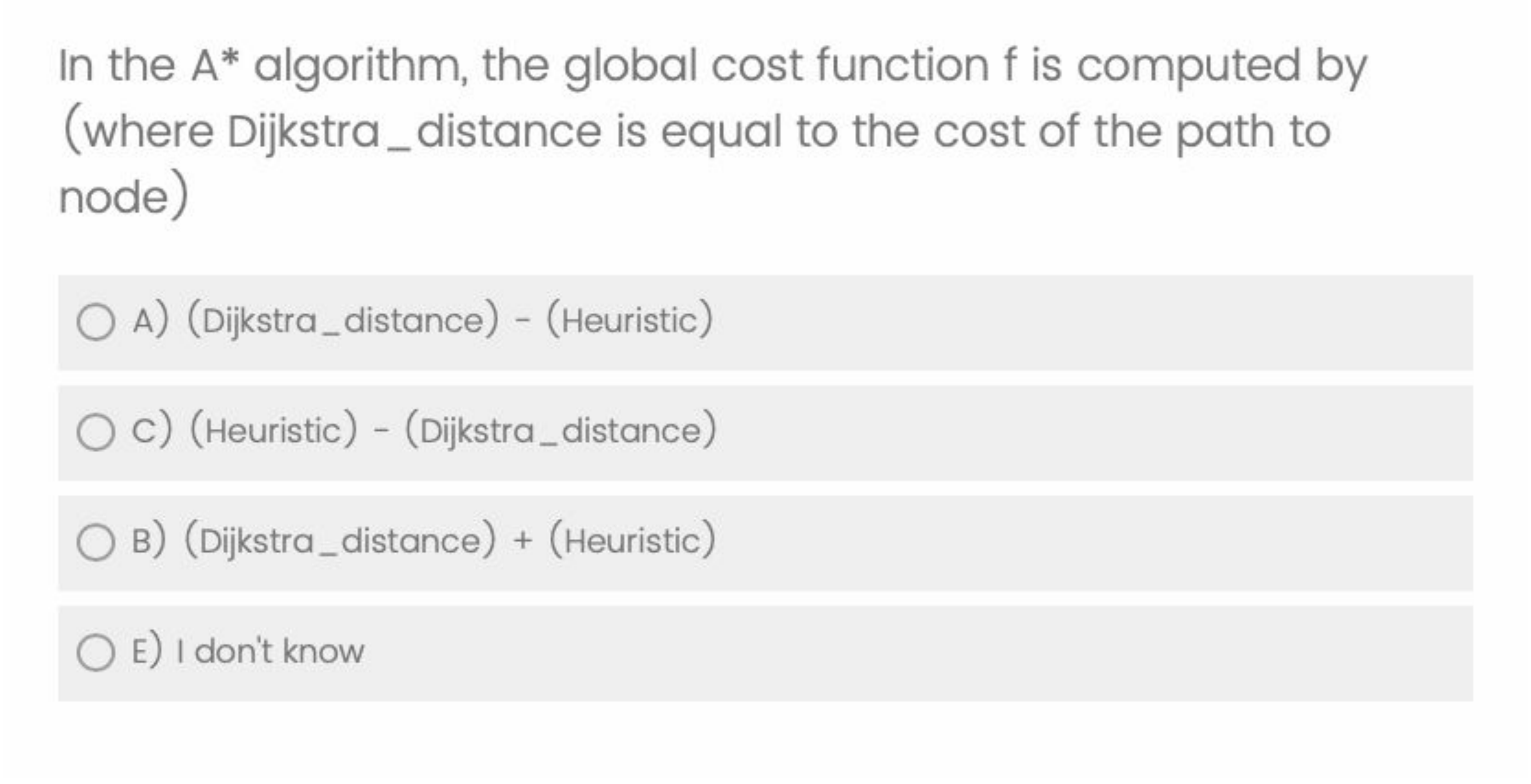
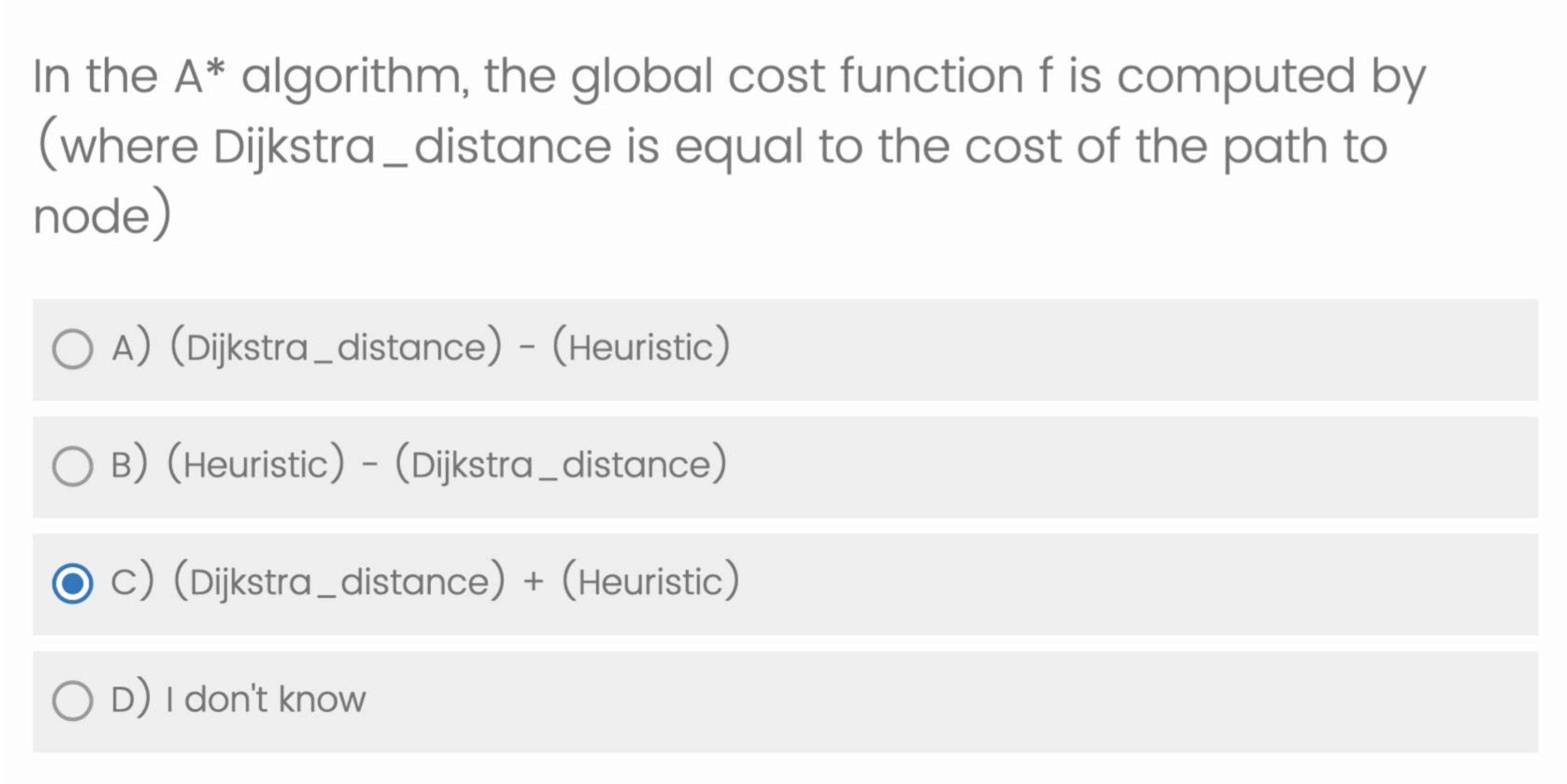


In the A* algorithm, the global cost function f is computed by (where Dijkstra_distance is equal to the cost of the path to node)

- ○ A) (Dijkstra_distance) - (Heuristic)
- ○ C) (Heuristic) - (Dijkstra_distance)
- ○ B) (Dijkstra_distance) + (Heuristic)
- ○ E) I don't know

Correction

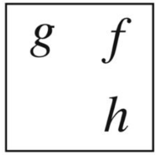


In the A* algorithm, the global cost function f is computed by (where Dijkstra_distance is equal to the cost of the path to node)

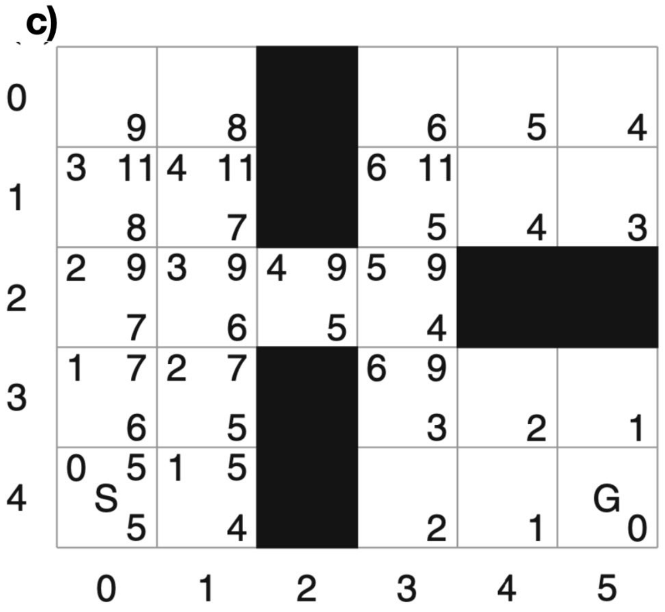
- A) (Dijkstra_distance) - (Heuristic)
- B) (Heuristic) - (Dijkstra_distance)
- [✓ C) (Dijkstra_distance) + (Heuristic)]
- D) I don't know

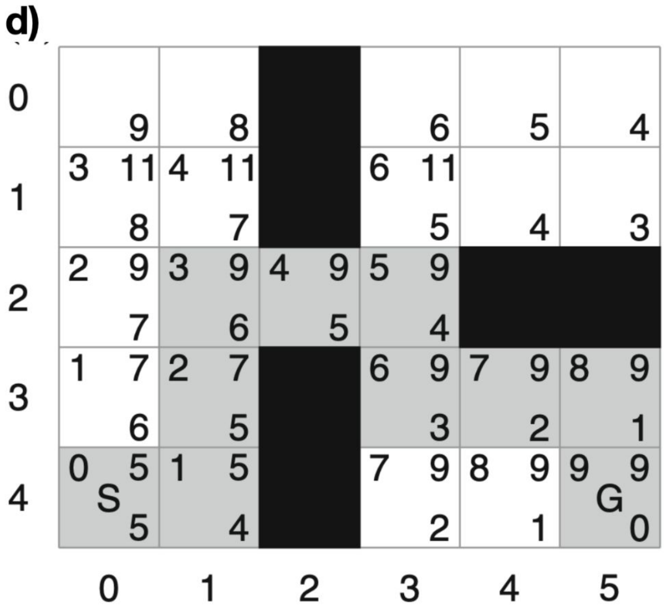
Correction: It is the definition of the A* algorithm

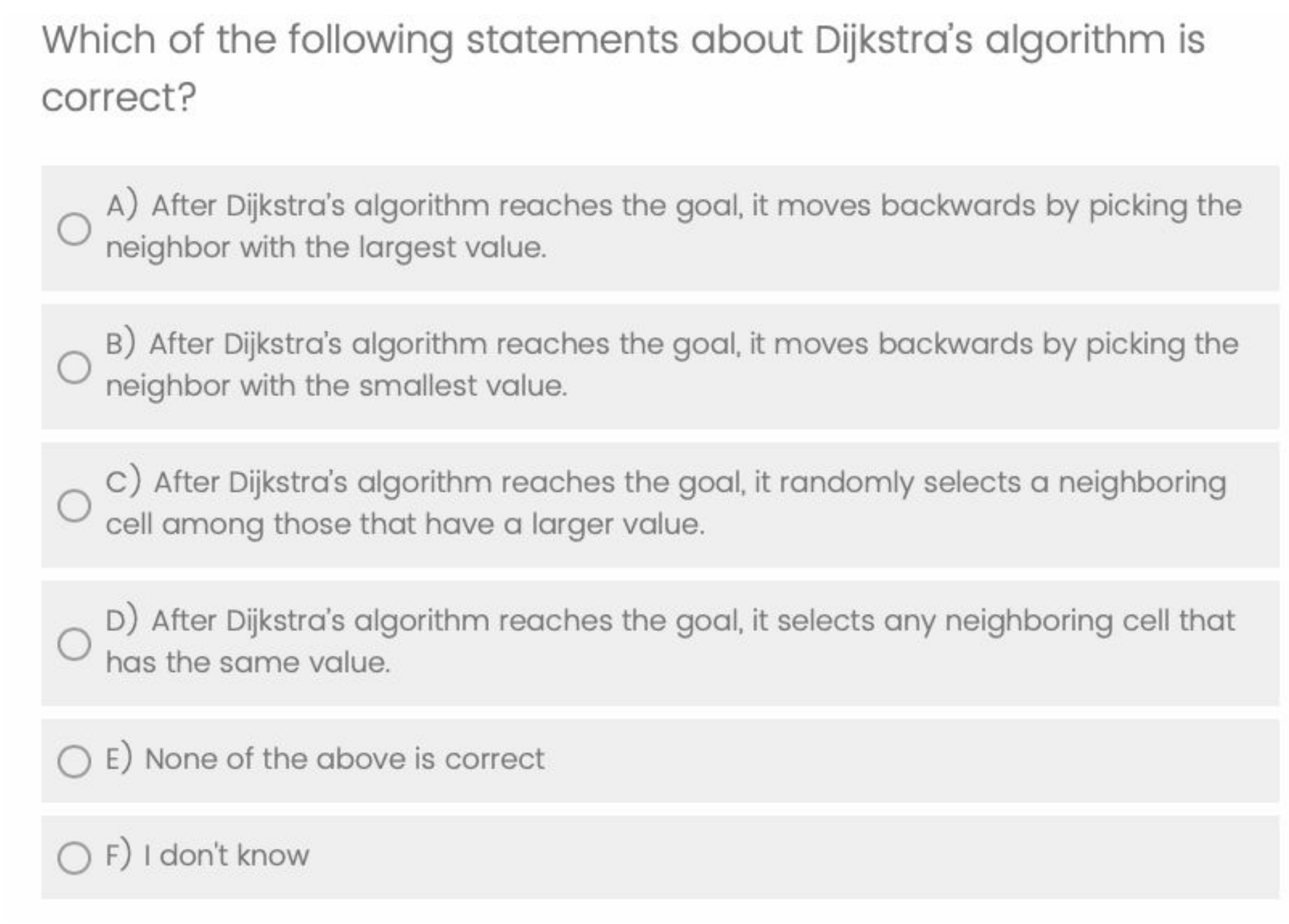
Correction



$$f(x,y) = g(x,y) + h(x,y)\,.$$

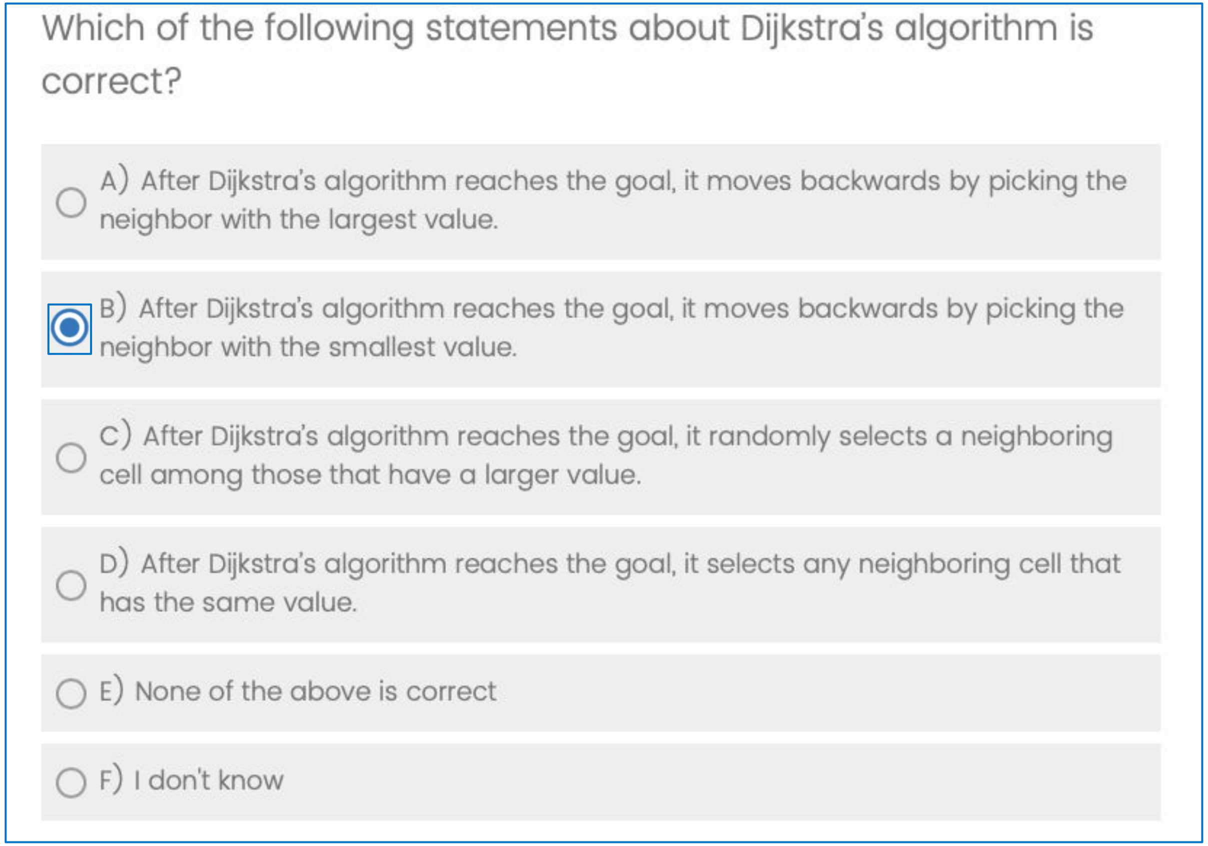






Which of the following statements about Dijkstra's algorithm is correct?

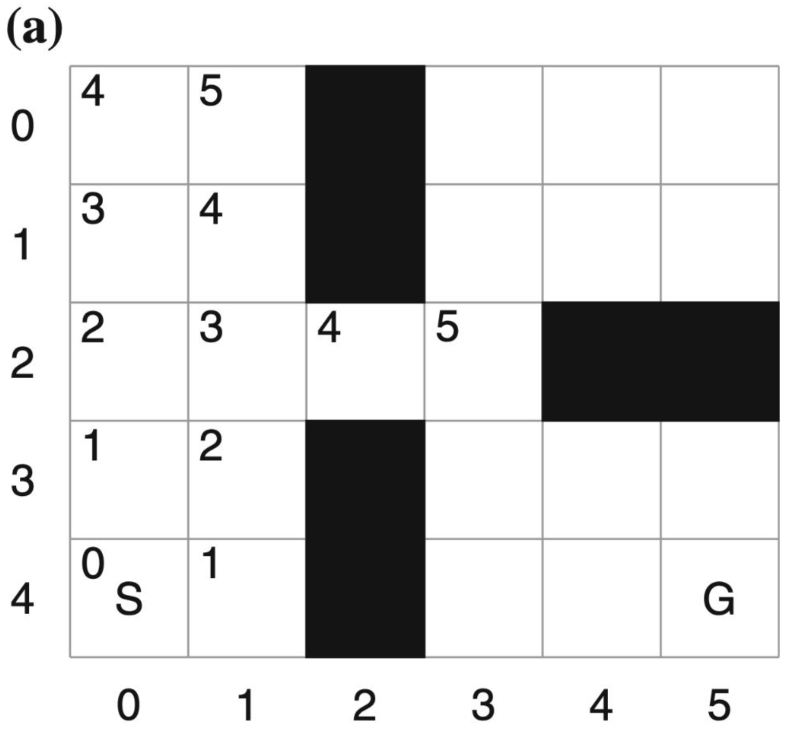
- ○ A) After Dijkstra's algorithm reaches the goal, it moves backwards by picking the neighbor with the largest value.
- ○ B) After Dijkstra's algorithm reaches the goal, it moves backwards by picking the neighbor with the smallest value.
- ○ C) After Dijkstra's algorithm reaches the goal, it randomly selects a neighboring cell among those that have a larger value.
- ○ D) After Dijkstra's algorithm reaches the goal, it selects any neighboring cell that has the same value.
- ○ E) None of the above is correct
- ○ F) I don't know

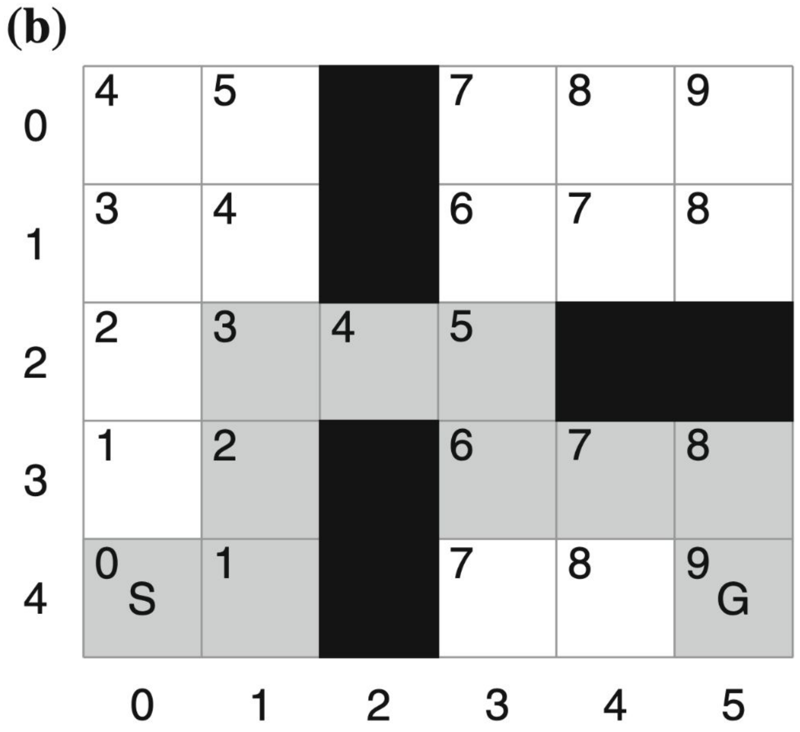

Which of the following statements about Dijkstra's algorithm is correct?

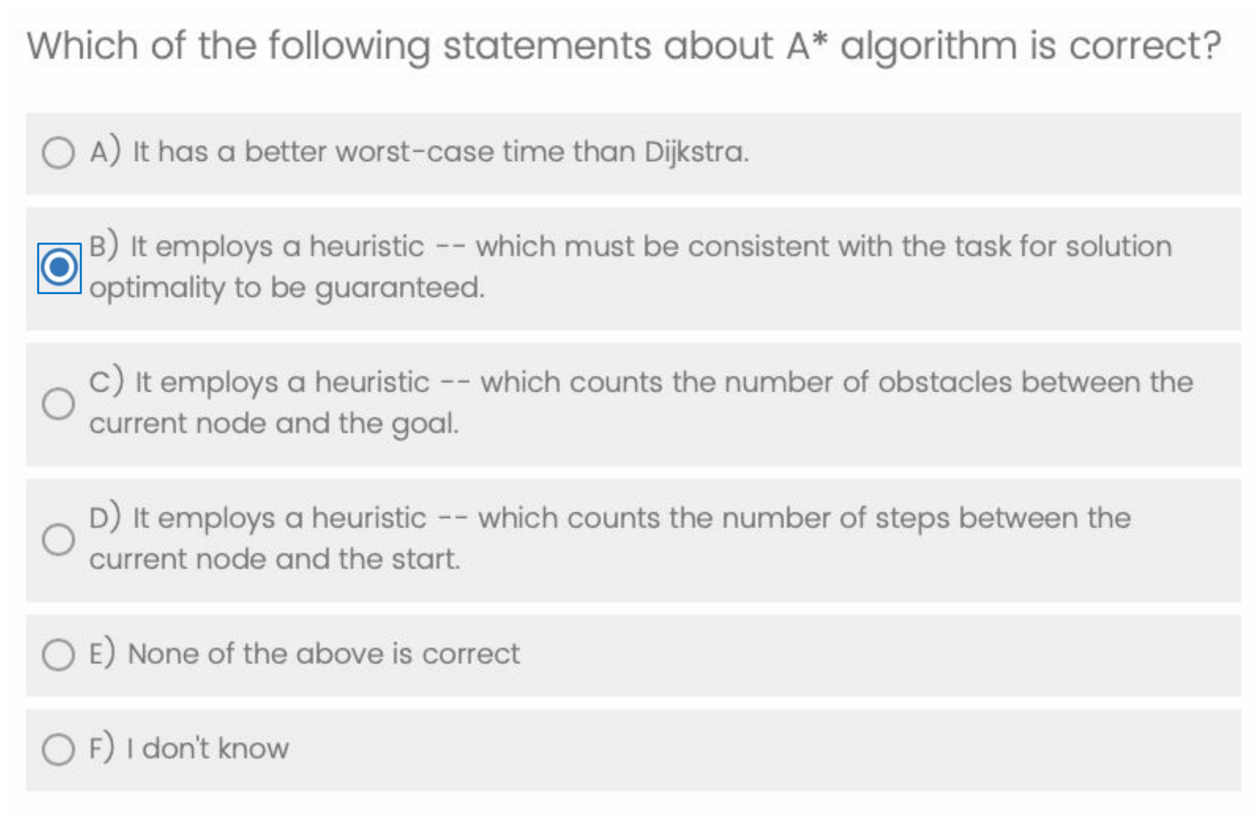
- ○ A) After Dijkstra's algorithm reaches the goal, it moves backwards by picking the neighbor with the largest value.
- ● B) After Dijkstra's algorithm reaches the goal, it moves backwards by picking the neighbor with the smallest value.
- ○ C) After Dijkstra's algorithm reaches the goal, it randomly selects a neighboring cell among those that have a larger value.
- ○ D) After Dijkstra's algorithm reaches the goal, it selects any neighboring cell that has the same value.
- ○ E) None of the above is correct
- ○ F) I don't know

Correction: Once the Dijkstra's algorithm reaches the goal, the path is simply traced back by following the neighbor with the smallest value (see next slide and see the pseudo code of Dijkstra).

Correction


(a)
0
1
2
3
4
4
5
3
4
2
3
4
5
1
2
0
S
1
G
0
1
2
3
4
5

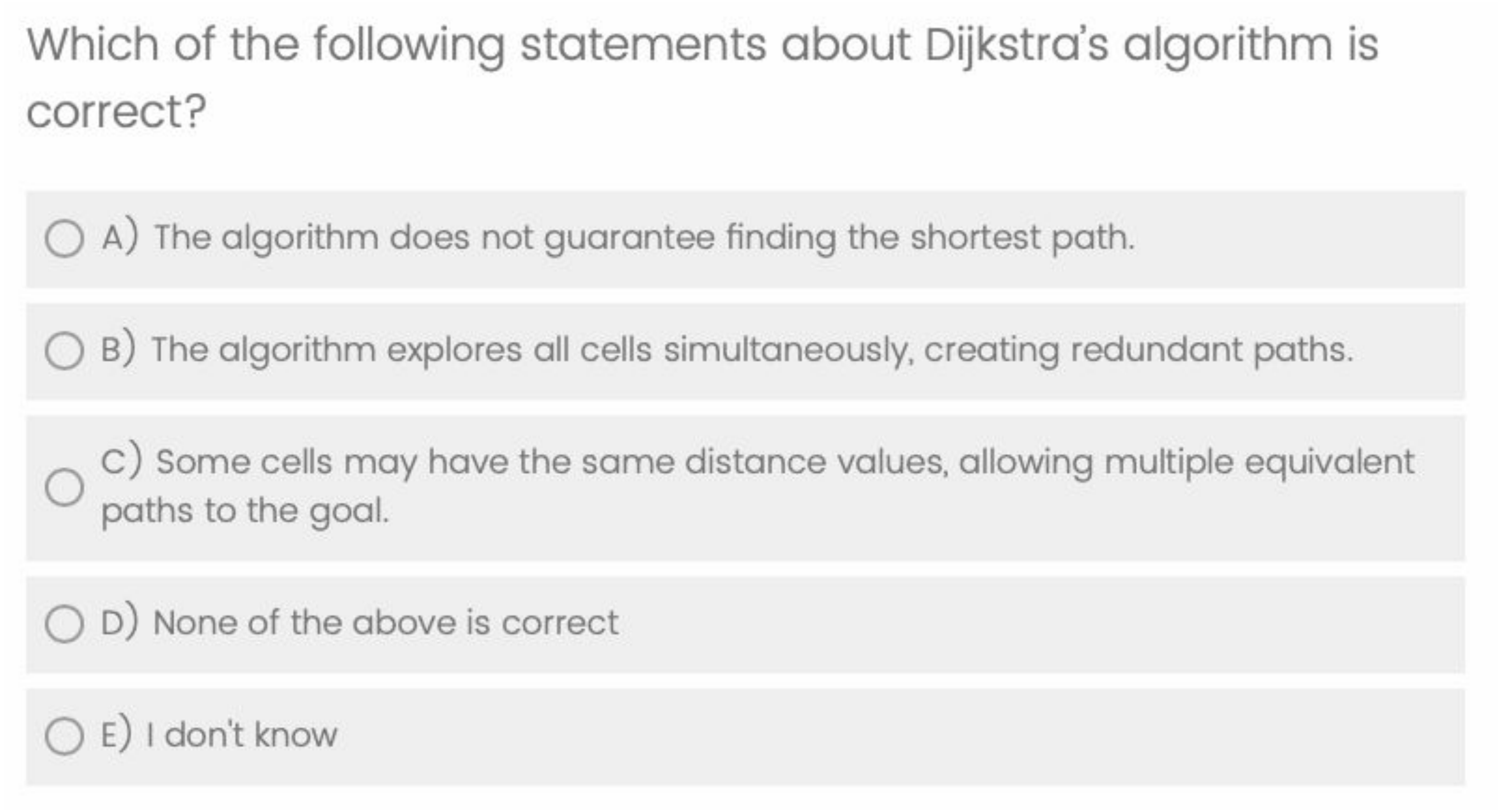

(b)
0
1
2
3
4
4
5
7
8
9
3
4
6
7
8
2
3
4
5
1
2
6
7
8
0
S
1
7
8
9
G
0
1
2
3
4
5



Which of the following statements about A* algorithm is correct?

- A) It has a better worst-case time than Dijkstra.
- [✓ B) It employs a heuristic -- which must be consistent with the task for solution optimality to be guaranteed.]
- C) It employs a heuristic -- which counts the number of obstacles between the current node and the goal.
- D) It employs a heuristic -- which counts the number of steps between the current node and the start.
- E) None of the above is correct
- F) I don't know

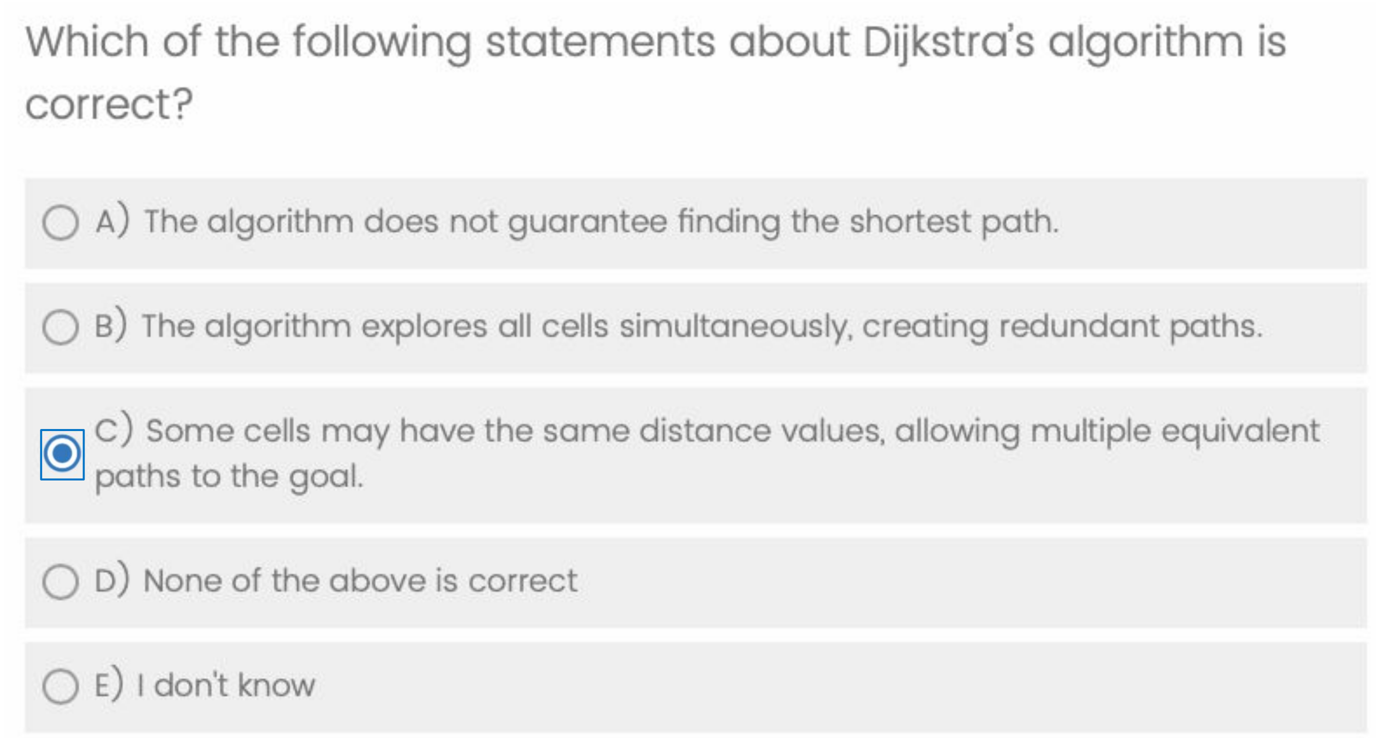
Correction: A* uses a heuristic function to estimate the cost from the current node to the goal. For the A* algorithm, the heuristic need to be both admissible (it never overestimates the true cost) and **consistent.**



Which of the following statements about Dijkstra's algorithm is correct?

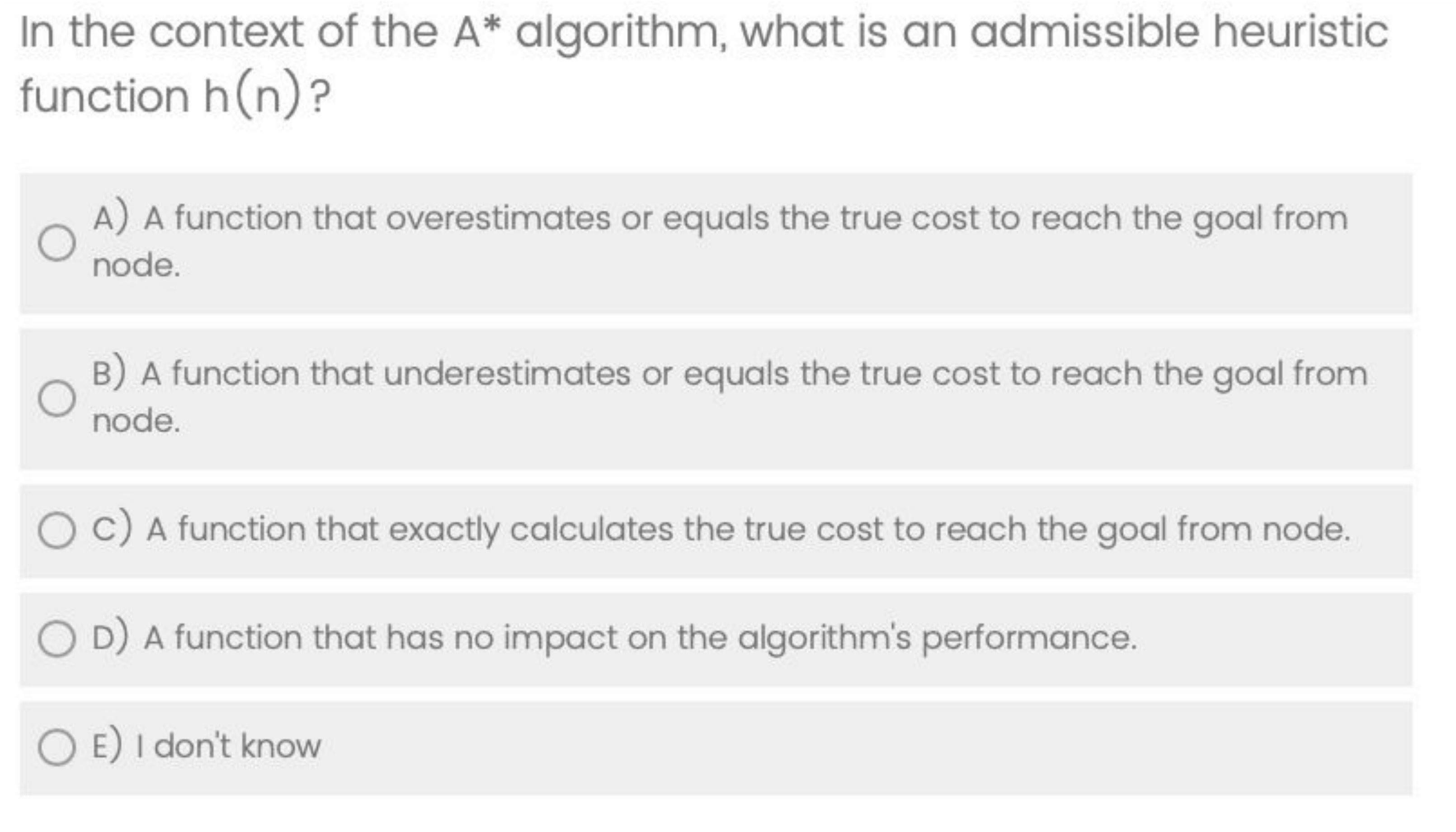
- A) The algorithm does not guarantee finding the shortest path.
- B) The algorithm explores all cells simultaneously, creating redundant paths.
- C) Some cells may have the same distance values, allowing multiple equivalent paths to the goal.
- D) None of the above is correct
- E) I don't know

Correction


Which of the following statements about Dijkstra's algorithm is correct?

- A) The algorithm does not guarantee finding the shortest path.
- B) The algorithm explores all cells simultaneously, creating redundant paths.
- [✓ C) Some cells may have the same distance values, allowing multiple equivalent paths to the goal.]
- D) None of the above is correct
- E) I don't know

Correction: Try the exercise on the implementation (,.ipynb); you can get multiple paths if you change some parameters…



In the context of the A* algorithm, what is an admissible heuristic function h(n)?

○ A) A function that overestimates or equals the true cost to reach the goal from node.

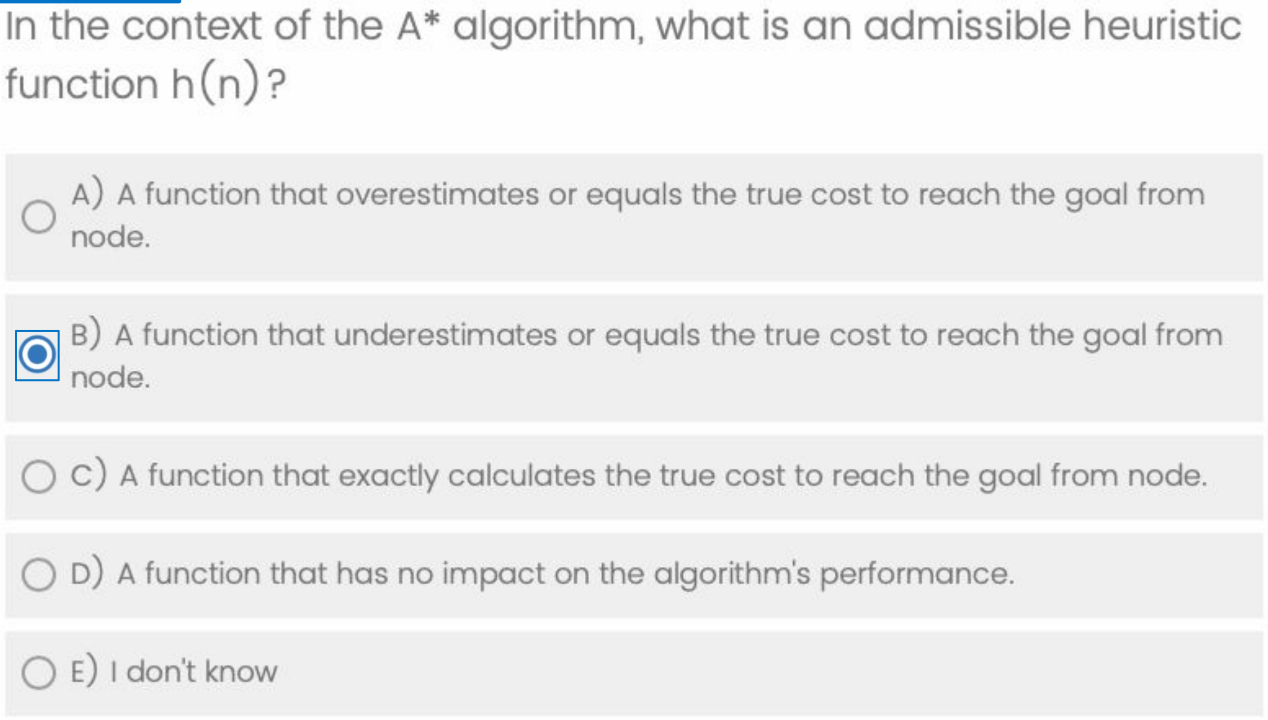
○ B) A function that underestimates or equals the true cost to reach the goal from node.

○ C) A function that exactly calculates the true cost to reach the goal from node.

○ D) A function that has no impact on the algorithm's performance.

○ E) I don't know

Correction

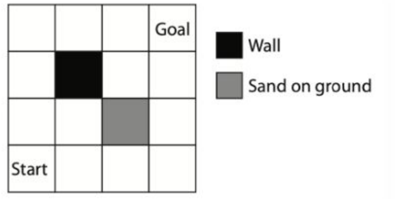


In the context of the A* algorithm, what is an admissible heuristic function h(n)?

- ○ A) A function that overestimates or equals the true cost to reach the goal from node.
- [✓ B) A function that underestimates or equals the true cost to reach the goal from node.]
- ○ C) A function that exactly calculates the true cost to reach the goal from node.
- ○ D) A function that has no impact on the algorithm's performance.
- ○ E) I don't know

Correction: In the A* algorithm, an admissible heuristic is one that never overestimates the true minimal cost from a node to the goal. This means the heuristic function h(n) satisfies h(n)≤h'(n) where h'(n) is the actual minimal cost from node n to the goal. By underestimating or exactly estimating the cost, the heuristic ensures that the A* algorithm finds the optimal path. (More details look on: **Admissible heuristic - Wikipedia**)

6. We apply Dijkstra's algorithm to the simple situation below to find the optimal path from start to goal. The wall cannot be passed, and the sand can be passed with some extra energy and time (cost of 2).

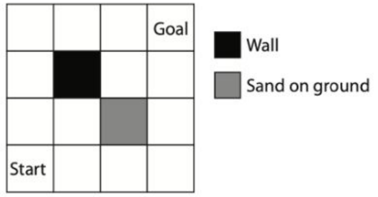


6.1) With a cost of 1 to move to any of the 4 neighbors (Manhattan) (excluding sand and walls), the number of optimal path(s) is/are:

- ◯ None
- ◯ 1 path
- ◯ 2 paths
- ◯ 3 paths
- ◯ 4 paths
- ◯ 5 paths
- ◯ I don't know

Correction

6. We apply Dijkstra's algorithm to the simple situation below to find the optimal path from start to goal. The wall cannot be passed, and the sand can be passed with some extra energy and time (cost of 2).

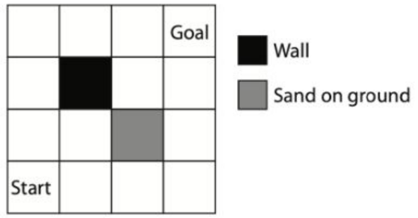


6.1) With a cost of 1 to move to any of the 4 neighbors (Manhattan) (excluding sand and walls), the number of optimal path(s) is/are:

- ○ None
- ○ 1 path
- [✓ 2 paths]
- ○ 3 paths
- ○ 4 paths
- ○ 5 paths
- ○ I don't know

Please try on the map

6. We apply Dijkstra's algorithm to the simple situation below to find the optimal path from start to goal. The wall cannot be passed, and the sand can be passed with some extra energy and time (cost of 2).

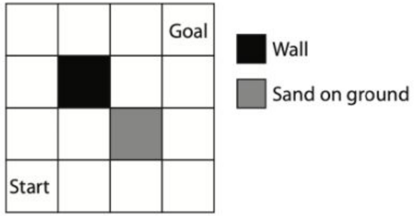


6.2) With a cost of 1 to move to any of the 8 neighbors (Euclidean) (excluding sand and walls), the number of optimal path(s) is/are:

- ○ None
- ○ 1 path
- ○ 2 paths
- ○ 3 paths
- ○ 4 paths
- ○ 5 paths
- ○ I don't know

Correction

6. We apply Dijkstra's algorithm to the simple situation below to find the optimal path from start to goal. The wall cannot be passed, and the sand can be passed with some extra energy and time (cost of 2).

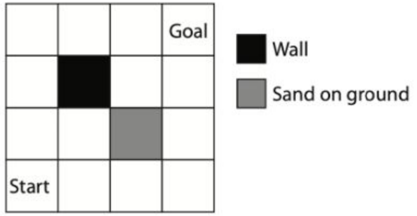


6.2) With a cost of 1 to move to any of the 8 neighbors (Euclidean) (excluding sand and walls), the number of optimal path(s) is/are:

- ○ None
- [✓ 1 path]
- ○ 2 paths
- ○ 3 paths
- ○ 4 paths
- ○ 5 paths
- ○ I don't know

Please try on the map

6. We apply Dijkstra's algorithm to the simple situation below to find the optimal path from start to goal. The wall cannot be passed, and the sand can be passed with some extra energy and time (cost of 2).

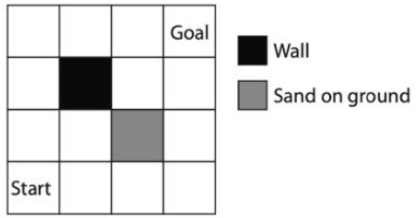


6.3) If you implemented A* and Dijkstra on the situation with 4 neighbors (Manhattan), what would you notice?

- A) A* gives the same result as Dijkstra for both the number of cells explored and the path.
- B) A* gives the same result as Dijkstra only for the path.
- C) A* gives the same result as Dijkstra only for the number of cells explored.
- D) None of the above answers is correct.
- E) I don't know

Correction

6. We apply Dijkstra's algorithm to the simple situation below to find the optimal path from start to goal. The wall cannot be passed, and the sand can be passed with some extra energy and time (cost of 2).



6.3) If you implemented A* and Dijkstra on the situation with 4 neighbors (Manhattan), what would you notice?

[✓ A) A* gives the same result as Dijkstra for both the number of cells explored and the path.]

B) A* gives the same result as Dijkstra only for the path.

C) A* gives the same result as Dijkstra only for the number of cells explored.

D) None of the above answers is correct.

E) I don't know

Please try on the map