DAT405 Assignment 8

Shivneshwar Velayutham

January 4, 2023

Problem 1a

In the worst case BFS might have to iterate through every node. At each level the max number of nodes = d^i with i starting at 0. The max number of leaves = d^{r-1} since r is the numbers of nodes in the shortest path to the goal/leaf. $Total = \sum_{i=0}^{r-1} d^i = \frac{d^r-1}{d-1}$ using the summation formula for a geometric progression.

Problem 1b

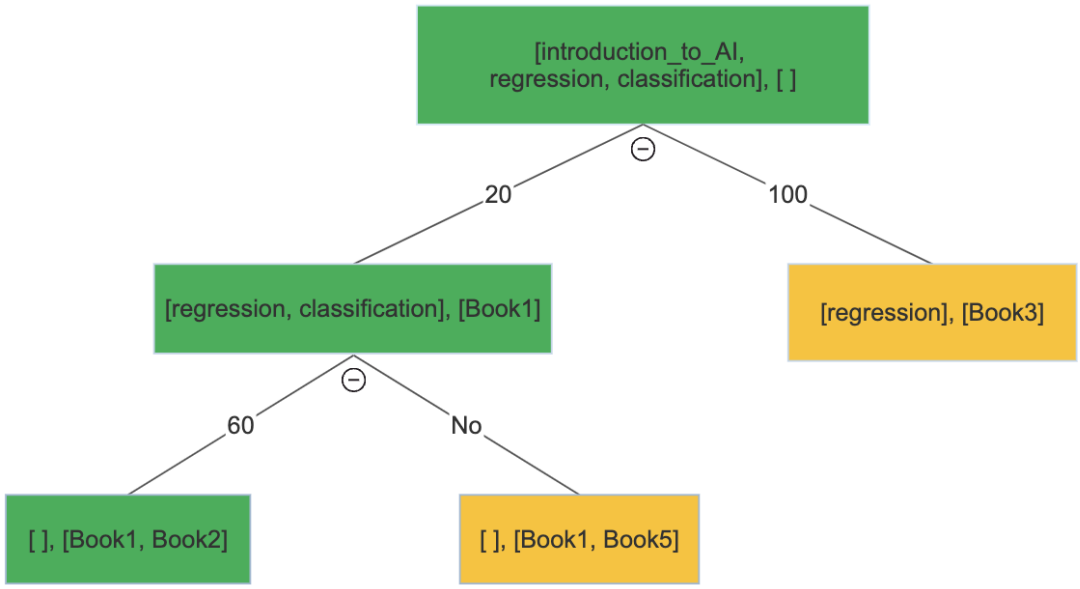
BFS will store all paths till depth r and each path will contain r number of nodes in it's path. The number of paths stored till depth r is equal to d^{r-1} since there will exist a path for each leaf. So the total number of units required is equal to $r \times d^{r-1}$.

Problem 2

When the upper node (ie. the node that lies above the line of nodes below) is set 1 then a loop ensues. This is because from the node next to 2 there exists two options, either to 2 or either go to the upper node. If the upper node's label is less than 2 then the loop begins since there is a triangular loop in the center of the graph which will never stop.

DFS can be modified such that it keeps tracks of the nodes already visited so as to avoid loops and not revisit nodes that have already been visited.

Problem 3a



Problem 3b

The heuristic function could the number of pages the book that has the smallest number of pages and contains the topic of the leftmost topic of the node. If the node does not have any topics then the heuristic function could return 0. This heuristic function is accurate since it tells us how much further pages we might need at the very least to at least study one more topic.

Problem 4a

Iterations below

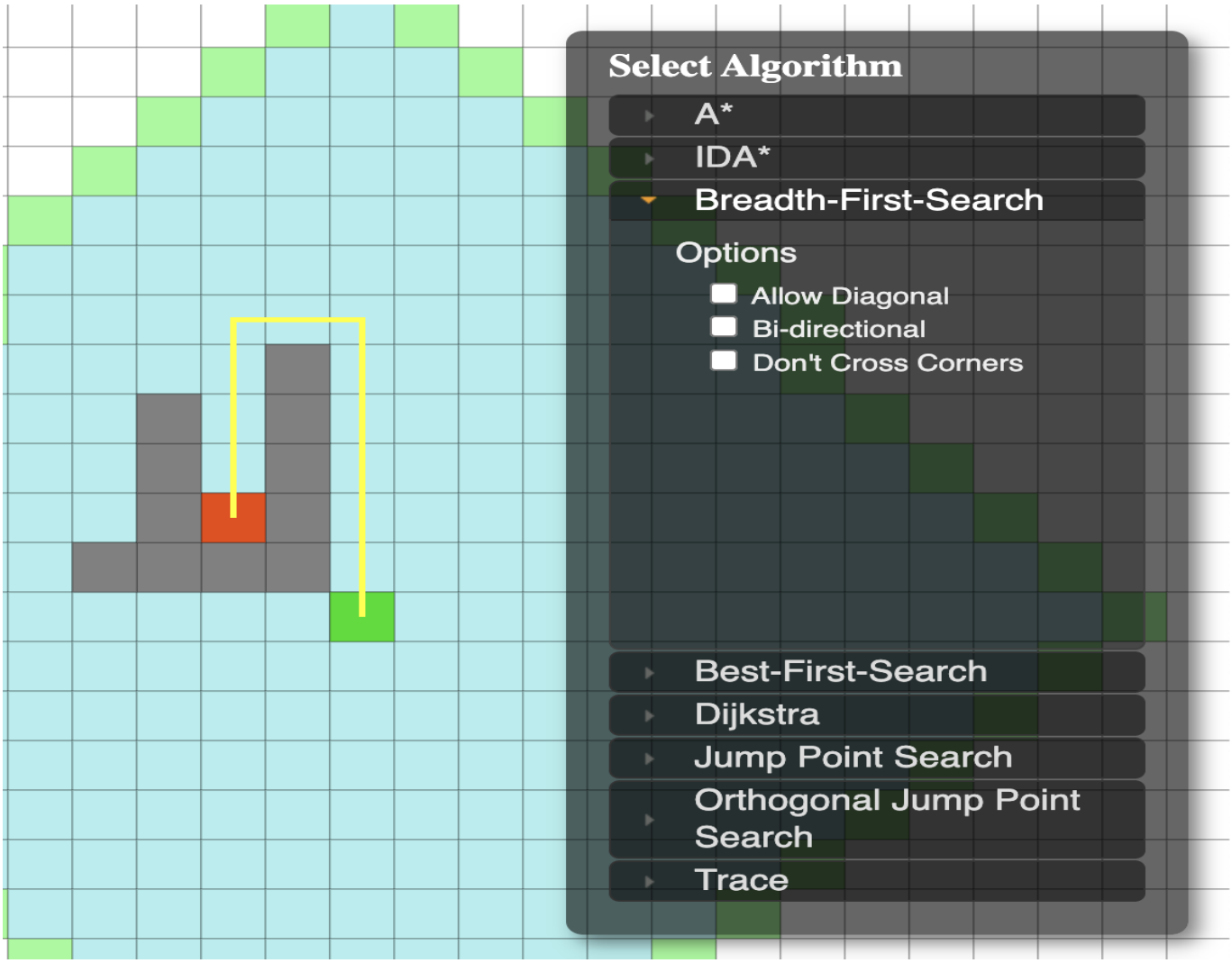
1. Starting with Path = s
2. Chosen node = 44 since it's the closest to the destination. Path = s, 44
3. Chosen node = 34 since it's the closest to the destination. Path = s, 44, 34
4. Since we've reached a dead end, a different path is chosen. The oldest path is chosen first as mentioned in the question. Chosen node = 42 since it was chosen first (it could've been 53 also). Path = s, 42
5. Chosen node = 32 since it's the closest to the destination. Path = s, 42, 32

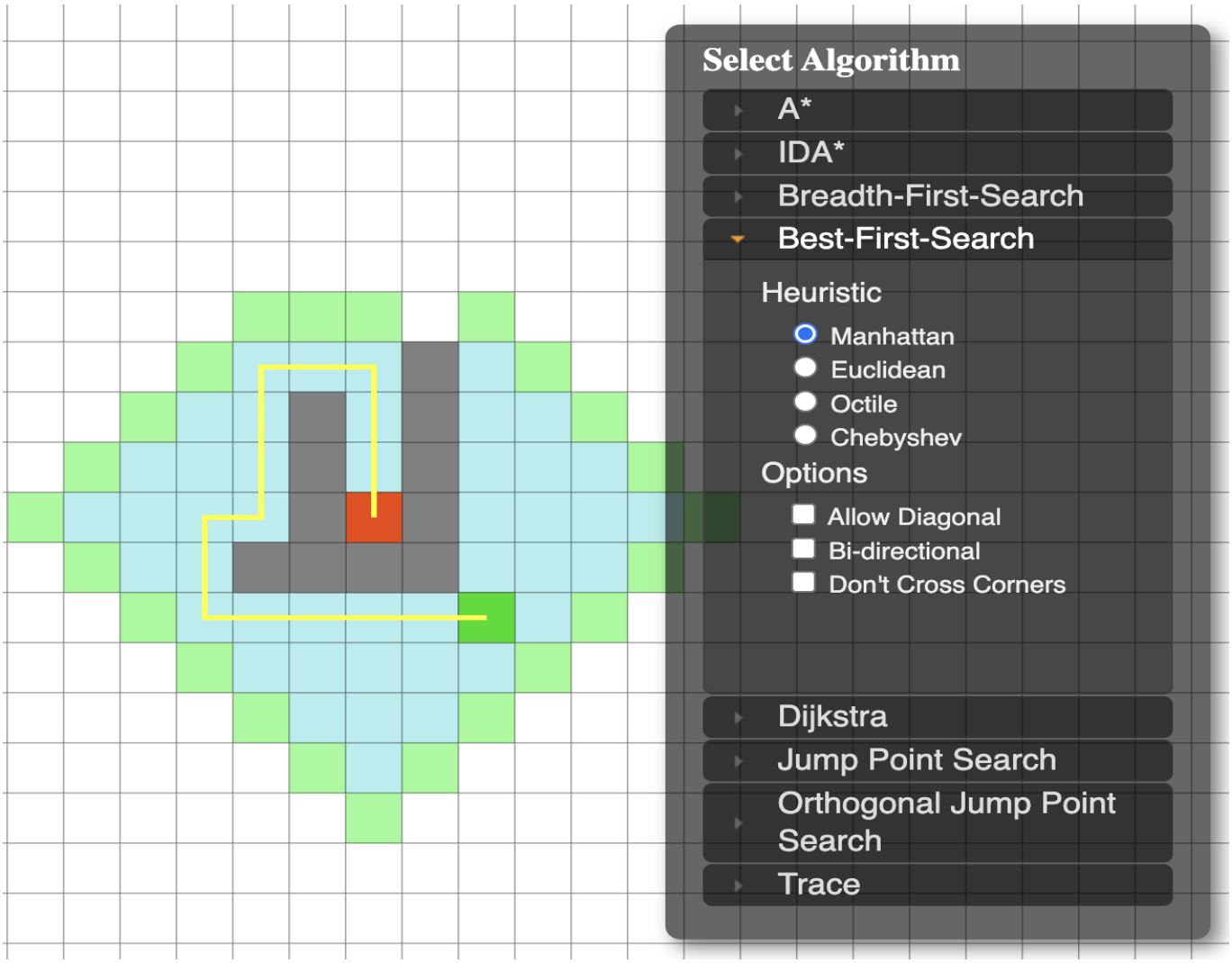
Problem 4b

1. BFS takes 364 operations and resulting path length is equal to 10. BFS is uninformed search and does not take into account any heuristics thus it takes longer as expected. It searches though the a large amount of the space until the destination is reached and chooses the shortest one from all paths to destination.
2. A* takes 71 operations and resulting path length is equal to 10. A* does much better since it uses Manhattan distance as the heuristic function therefore reducing the number of paths taken in account by not considering the ones that are further away from the destination.
3. BestFS takes 48 operations and resulting path length is equal to 10. BestFS is also an informed search like A* and uses a heuristic function so it does better than BFS. The difference between A8* and BestFS is that BestFS does not store the path but makes decisions purely based on the heuristic score. So when using A*, since it stores the cost of the path take, it backtracks and checks other paths whose cost are lesser but since BestFS does not take path cost into account it doesn't backtrack and just tries to reach the destination. Since BestFS does not take into account the path cost, it may not always result in optimal path but in our current scenario it results in the optimal path.

Problem 4c

As seen below we can see the BestFS takes a longer route than BFS which gives us the optimal path.


Select Algorithm
A*
IDA*
Breadth-First-Search
Options
Allow Diagonal
Bi-directional
Don't Cross Corners
Best-First-Search
Dijkstra
Jump Point Search
Orthogonal Jump Point Search
Trace



As discussed earlier, BestFS does not take into consideration the cost of the path but only the heuristic of each state/position along the path. So it will choose a path containing nodes closer to the destination even if it results in a longer path. Whereas BFS considers the entire space and even though it takes a long time it will give us the optimal path to destination since it does not discount/remove any paths from consideration.

Problem 5a

Normally Markov decision processes are non deterministic and there exists probabilities for state transitions. The generic search problem can be described as a deterministic Markov decision process ie. the probability of each transition from one state/position to another state/position is 1.

Problem 5b

The big disadvantage of value iteration algorithm is that it calculates for the entire environment thus making it much slower than A* algorithm that discounts/does not take into consideration states/positions in the environment which will not help in finding the optimal path as they're too far away.

The advantage of value iteration algorithm is that it's possible to set the value for gamma which tells the algorithm whether to fine tune for immediate rewards or later rewards. Since gamma is a value between 0 and 1 we can set fine tune based on what kind of search scenario we have. At the same time this could be a disadvantage when a bad gamma value has been chosen so care must be taken when selecting the value.