DAT405/DIT407 Introduction to Data Science and AI, LP2 2022

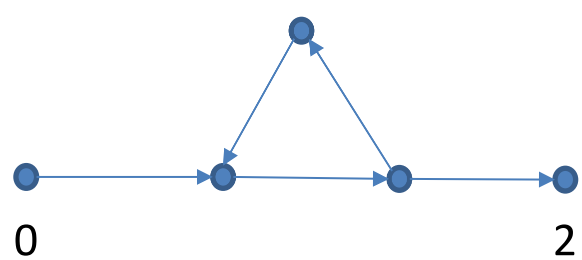
Assignment 8: Rule-based AI

This assignment is about problem solving only. There is no programming assignment. Your answers should be given as a written report. Answer to each question with adequate discussion/justification/analysis.

1) [2p] The branching factor '*d*' of a directed graph is the maximum number of children (outer degree) of a node in the graph. Suppose that the shortest path between the initial state and a goal is of length '*r*'.

> a) What is the maximum number of Breadth First Search (BFS) iterations required to reach the solution in terms of '*d*' and '*r*'?
>
> b) Suppose that storing each node requires one unit of memory and the search algorithm stores each entire path as a list of nodes. Hence, storing a path with *k* nodes requires *k* units of memory. What is the maximum amount of memory required for BFS in terms of '*d*' and '*r*' ?

2) [1p] Take the following graph where 0 and 2 are respectively the initial and the goal states. The other nodes are to be labelled by 1,3 and 4.



Suppose that we use the Depth First Search (DFS) method and in the case of a tie, we chose the smaller label. Find all labelling of these three nodes, where DFS will never reach to the goal! Discuss how DFS should be modified to avoid this situation?

3) [2p] A publisher allows teachers to "build" customised textbooks for their courses by concatenating the text from different books in their catalogue. The catalogue contains the following books, together with the number of pages that the book contains, and the topics covered in that book:

Books in catalogue	Number of pages	Topics covered
book1	20	[introduction_to_AI]
book2	60	[regression, classification]
book3	100	[introduction_to_AI, search, classification]
book4	80	[machine_learning, neural_networks]
book5	100	[regression, classification]

Suppose we define a node to be a pair *(Topics, Books)* where *Books* is a list of the books that will make up a customised textbook and *Topics* is a list of topics that must be covered by the customised textbook but are not already covered by *Books*. Therefore, a node is only valid if none of the books in *Books* covers any of the topics in *Topics*.

Child nodes are obtained by selecting a topic from *Topics*, then selecting a book that covers this topic, then adding this book to *Books* and finally removing all of the topics that are covered by this book from the *Topics* list. For example, if we have the node *([introduction_to_AI, classification],[])* and we select the topic 'introduction_to_AI' then the child nodes will be *([],[book3])* and *([classification],[book1])*. Thus, each arc in the graph adds one book which covers one or more topics. Suppose that the cost of an arc is equal to the number of pages in the selected book.

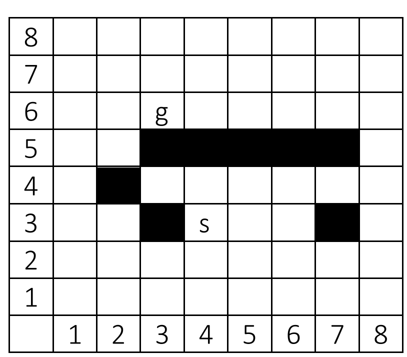
The goal is to design a customised textbook that covers all of the topics requested by the teacher, i.e., the topics in list *Topics*. The start node is *(Topics,[])* and the goal nodes have the form *([],CustomisedTextbook)*, where *CustomisedTextbook* is a list of books selected from the catalogue. The cost of the path from the start node to a goal node is equal to the total number of pages in the customised textbook, and an optimal customised textbook is one that covers all of the requested topics (i.e., all of the topics in *Topics*) with the fewest pages.

a) Suppose a teacher requests a customised textbook that covers the topics *[introduction_to_AI, regression, classification]* and that the algorithm always selects the leftmost topic when generating child nodes of the current node. Draw (by hand) the search space as a tree expanded for a lowest-cost-first search, until the first solution is found. This should show all nodes expanded. Indicate which node is a goal node, and which node(s) are at the frontier when the goal is found.

b) Give a non-trivial heuristic function *h* that is admissible. [*h(n)=0* for all *n* is the trivial heuristic function.]

4) [3p] Consider the problem of finding a path in the grid shown below from the position *s* to the position *g*. A piece can move on the grid horizontally or vertically, one square at a time. No step may be made into a forbidden shaded area. Each square is denoted by the xy coordinate. For example, s is 43 and g is 36. Consider the Manhattan distance as the heuristic. State and motivate any assumptions that you make.

a) Write the paths stored and selected in the first five iterations of the A* algorithm, assuming that in the case of tie the algorithm prefers the path stored first.



b) Solve this problem using the software in http://qiao.github.io/PathFinding.js/visual/
Use Manhattan distance, no diagonal step and compare A*, BFS and Best-first search. Describe your observations. Explain how each of these methods reaches the solution. Discuss the efficiency of each of the methods for this situation/scenario.

c) Using a board like the board used in question 4a) or in http://qiao.github.io/PathFinding.js/visual/, describe and draw a situation/scenario where Breadth-first search would find a shorter path to the goal compared to Greedy best-first search. Consider that a piece can move on the grid horizontally or vertically, but not diagonally. Explain why Breadth-first search finds a shorter path in this case.

5) [2p] This question is about the relation of Markov decision processes in Assignment 5 and search algorithms.

a) Discuss when and how the generic search problem can be described as a Markov decision process (MDP).
b) When the search problem can be written as an MDP, what are the advantages and disadvantages of the value iteration algorithm over the A* algorithm?

What to submit

- A pdf report stating
    - Your names and how many hours each person spent on the assignment.
    - Answers to questions, and discussion/motivation of each answer.

Make sure to give the names of all the people in the group on all files you submit!

Deadline: Wednesday 4 January 2023 at 23:59.