Inoussa Adouba
Matrikelnummer: 01207837

Assignment 3

1. (a)

Initial state: each set of pieces not being bound together.

States: Different pieces being put together and not overlapping at each added piece

Actions: connect pieces together or disconnect, no overlapping

Goal Test: no loose ends and no overlapping tracks

Path Cost: 1 for each action

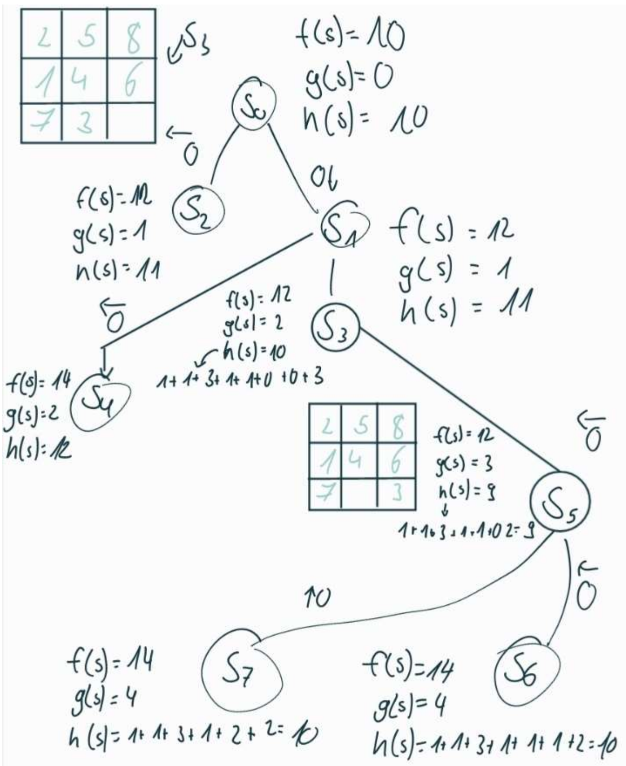
(b)

Depth First Search would be suitable for this Search Problem as the Agent would first try to put all pieces together and if it doesn't fit the agent would save all moves, he did up to that state where he lost and start again, still knowing that at a specific level a specific state wasn't the solution. Also, since if the branching factor is higher BFS Algorithm would render inefficient as it consumes space and time. Also, DFS is memory efficient which would also result in it being an optimal algorithm to use.

(c)

Each of the supplied parts is needed to build a fully finished structure and removing a "fork" piece would result in a curve that can no longer be connected. A dead-end would appear which will make the problem unsolvable.

2.



3.

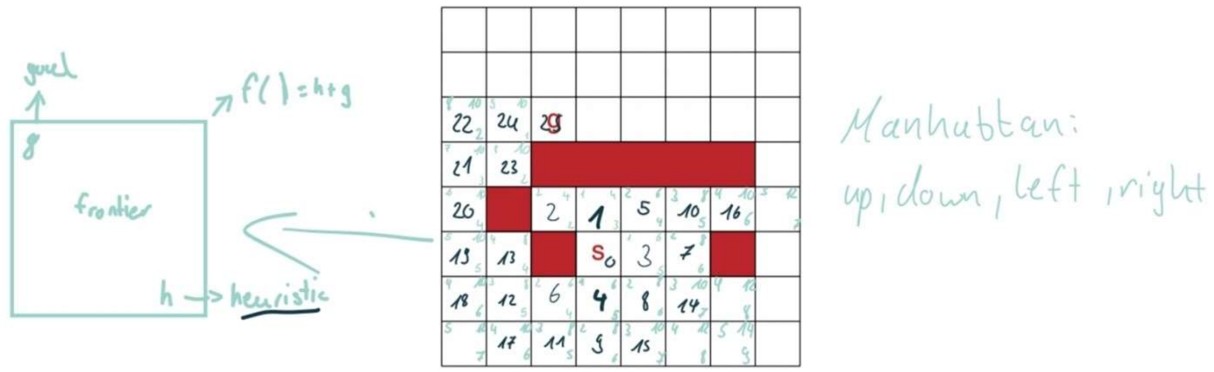
Uniform cost Search is indeed a version of A* Search since uniform cost search uses the maximum priority based on the optimal next step with the lowest cost, however A* Search uses the maximum

https://www.educative.io/answers/what-is-uniform-cost-search

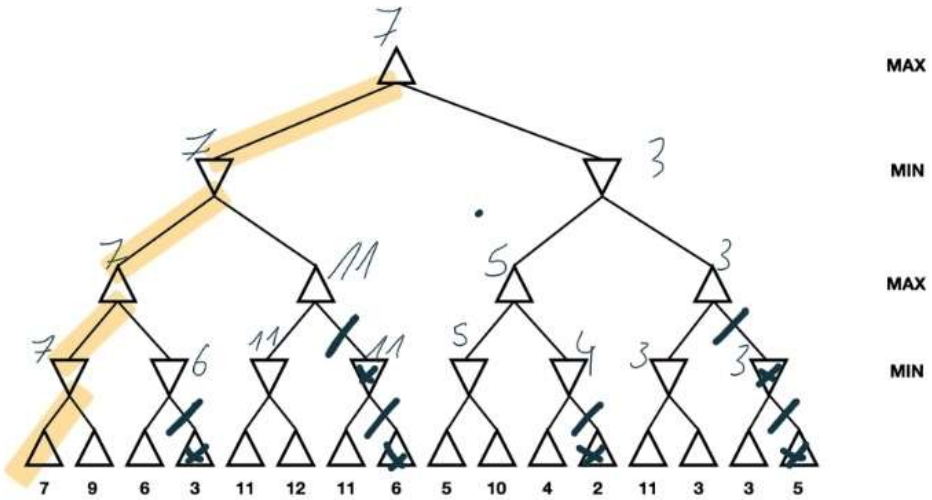
Inoussa Adouba
Matrikelnummer: 01207837

priority based on the heuristic to determine the optimal next step. When using the uniform cost search algorithm, the heuristic function is not required to determine the next step so instead of each optimal step being evaluated as f() = g() + h() which is the case in A* Search the optimal step will be evaluated as f() = g().

4.



5.



6.

Since the suboptimal player min doesn't choose the worst value possible for the max player the utility of the max player can't be lower than the utility for the optimal min player because the optimal min player chooses the lowest value possible and the suboptimal doesn't do so always. The minimax algorithm always assumes rational optimal play any deviation from the optimal min player to a suboptimal min player can only result in benefit for the Max player.