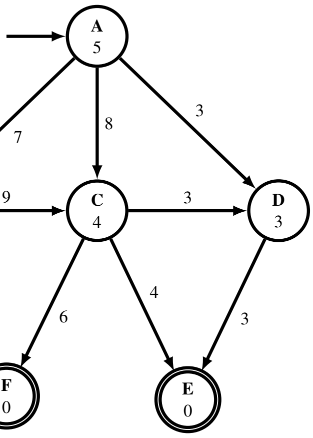
Introduction to Artificial Intelligence

Exercise Sheet 3

Handed out: 04.12.2020 Prof. Dr. Susanne Biundo-Stephan
Due: 18.12.2020 Conny Olz

Exercise 1 (6 points)

Given below is a graphical representation of a state space. Nodes represent states and edges represent actions. Nodes are labeled with a name and the estimated costs to reach a goal from the node n (its heuristic value, denoted $h(n)$). Edges are labeled with the costs of the underlying action. The initial state is A. Goal states are F and E.



Please provide the sequence of *expanded* nodes (including the returned goal node) for the given search strategies. Further provide the solution sequence and the costs of this solution.

Use a standard queue for the BFS. If you have more than one successor node, insert them in alphabetic ordering (A before B, B before C, and so on).

For the other search strategies, use a *priority queue*, ordered according to the evaluation function f of the given strategy (e.g. with $f(n) = h(n)$ for a greedy search). If there is more than one node with the same f, use alphabetic ordering.

Hand in your results in Moodle. Please submit just one solution per group and make sure you submitted it properly. You can review your achieved score after the deadline.

1. BFS as graph search.

   Expanded: A __ __ __ __ __ __ __ __ __

   Solution path: A __ __ __ __ __ __ __ __ __ Costs: ____

2. Uniform cost search as tree search.

   Expanded: A __ __ __ __ __ __ __ __ __

   Solution path: A __ __ __ __ __ __ __ __ __ Costs: ____

3. Greedy search as graph search.

   Expanded: A __ __ __ __ __ __ __ __ __

   Solution path: A __ __ __ __ __ __ __ __ __ Costs: ____

4. A* search as tree search.

   Expanded: A __ __ __ __ __ __ __ __ __

   Solution path: A __ __ __ __ __ __ __ __ __ Costs: ____

Introduction to Artificial Intelligence

Exercise Sheet 3

Handed out: 04.12.2020 Prof. Dr. Susanne Biundo-Stephan
Due: 18.12.2020 Conny Olz

Exercise 2 (5 points)

Implement an A^* *search as graph search* that outputs the expanded search nodes for a given state space. The state space is given in form of a graph and – as usual – passed via the *stdin*. Nodes represent states and edges represent actions.

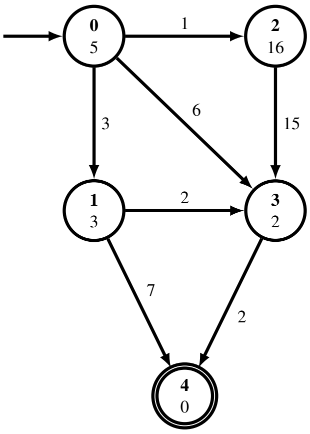
Nodes are labeled with the estimated costs to reach a goal state (its heuristic value), edges with the costs of the action underlying the edge.

A graph representing the state space with n nodes and m edges is encoded as follows:

- The first line holds the values n and m as Integer values (separated by a space).
- Line 2 to $n+1$ (i.e. the next n lines) provide the heuristic values of he nodes 0 to $n-1$. The first node represents the initial state of the problem, the last one the (only) goal state.
- Line $n+2$ to $n+1+m$ (i.e. the next m lines) each represent a labeled edge. Each consists of three Integer values. The first two Integers represent the start and end node of the edge, the third Integer its costs.

The following figure gives a state space as well as its encoding:

```
5 7
5
3
16
2
0
0 1 3
0 2 1
0 3 6
1 3 2
1 4 7
2 3 15
3 4 2
```



Please output the list of expanded nodes (one per line). Getting the graph given above as input, your program should e.g. output the following sequence (more test cases at the DomJudge server):

```
0
1
3
4
```

Be aware that the goal node is contained in the list (though it is technically not expanded).