10. Exercise sheet, Block 2
A* algorithm

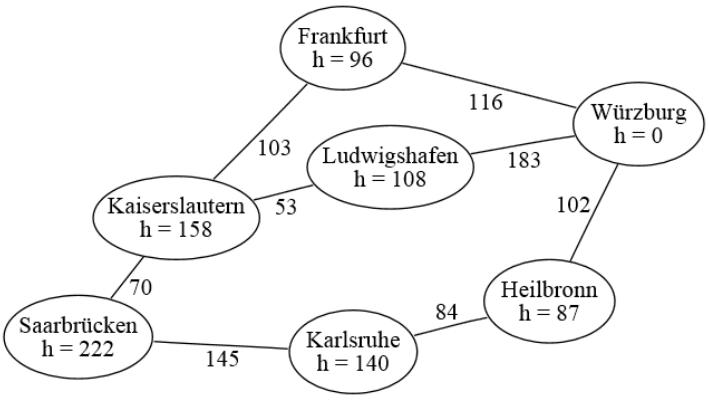

Figure 1: source: https://de.wikipedia.org/wiki/A*-Algorithmus

In this exercise the A* algorithm should be implemented with the example from Fig. 1. You should implement the algorithm in C++. Provided are templates with "Node" class, "AStar" class and "main" function. The "AStar" class already contains "Open" and "Close" lists as well as the function that can be used for sorting, since priority queue structure doesn't exists in pure C++. Your task is to complete the function "AStar::computePath()" which must correctly find a shortest path from "start" to "target" Nodes. If necessary you may add additional functions to make the code elegant. In any case the program must be compilable, runnable and produce the output, otherwise the code will be not accepted.

The function value for A* can be computed as:

$$f(n) = g(n) + h(n) \tag{1}$$

with the cost g(n) from the start to the node n and the heuristic value h(n) for the node n. Values for 'g' and 'h' are provided (see Fig. 1).

Contact: Dzmitry.Stsepankou@medma.uni-heidelberg.de

Detailed description

(a) Create the nodes and edges according to Fig. 1 (Already implemented in AStar default constructor).

(b) Implement the A* algorithm. Compute and print the shortest path from Saarbrücken to Würzburg (Complete "AStar::computePath()"). During the computation display the open list entires after each update of the list (If you add neighbors of one node to the open list you can display the open list entires after adding all of the neighbors).

Contact: Dzmitry.Stsepankou@medma.uni-heidelberg.de