Foundations of Artificial Intelligence

M. Helmert, M. Wehrle
T. Keller
Spring Term 2016

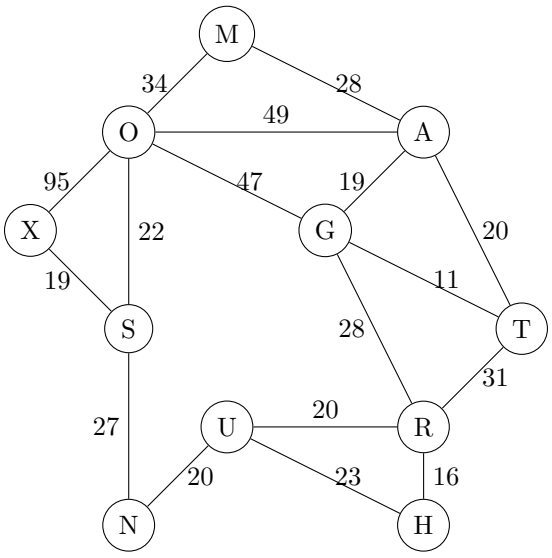
University of Basel
Computer Science

Exercise Sheet 5

Due: April 8, 2016

Exercise 5.1 (3+3 marks)

Consider the following map:



Let the air-line distance between Atlantis (A) and the other cities be given by the following table:

city	distance
A	0
G	14
H	50
M	25
N	36
O	42
R	38
S	42
T	13
U	51
X	47

Consider the heuristic that maps each state to its air-line distance to Atlantis (A).

(a) Provide the search tree of A* (without reopening) when queried for the shortest path from Hel (H) to Atlantis (A). Indicate the order in which nodes are expanded and annotate each node with its f-, g-, and h-values.

(b) Provide the search tree of greedy best first search (without reopening) when queried for a path from Hel (H) to Atlantis (A). Indicate the order in which nodes are expanded. Compare the result to the result of (a).

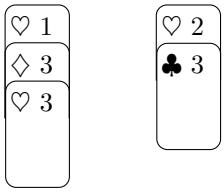
Exercise 5.2 (2+3+1 marks)

The task in this exercise is to write a software program. We expect you to implement your code on your own, without using existing code you find online. If you encounter technical problems or have difficulties understanding the task, please let us – Patrick Buder or Thomas Keller – know *sufficiently ahead of the due date*.

(a) We have extended the interface `StateSpace` with a method that returns a heuristic value for the given state (the method is called `public int h(State s)`). The `FreecellStateSpace` class already contains the skeleton of this method. Implement a heuristic that computes the sum of

- the number of cards that have not been moved onto a foundation pile yet, and
- the number of cards that have to be moved to another (non-foundation) pile at least once because there is a card of the same suit and lower rank below it in the same pile.

To illustrate the heuristic, consider the following example state s for a Freecell instance with cards up to rank 3:



All cards that are not depicted have already been moved to the corresponding foundation pile. The first part of the proposed heuristic counts the number of cards that are not yet on a foundation pile, which evaluates to 5. The second part evaluates to 1, as there is only one card (♡ 3) that has to be moved onto another non-foundation pile because there is a card of the same suit and lower rank below it (♡ 1). The heuristic value of s is therefore $h(s) = 6$.

(b) Implement A* without node reopening (please use the class skeleton that can be found on the website). Make sure that the value of the member variable `expandedStates` of the parent class `SearchAlgorithmBase` is updated correctly.

(c) Test your implementation on the example problem instances you can find on the website. Set a time limit of 10 minutes and a memory limit of 2 GB for each run. On Linux, you can set a time limit of 10 minutes with the command `ulimit -t 600`. Running your implementation (in a file called UniformCostSearch.java) on the first example instance with

java -Xmx2048M AstarSearch freecell freecell_inst_1

sets the memory limit to 2 GB. If the RAM of your computer is 2GB or less, set the memory limit to the amount of available RAM minus 256 MB instead and describe in your solution how much RAM was used.

Report runtime and number of node expansions for all instances that can be solved within the given time and memory limits. For all other instances, report if the time or the memory limit was violated.

The exercise sheets can be submitted in groups of two students. Please provide both student names on the submission.