Lab class: uninformed and informed search

Steven Schockaert

Week 3

At the end of the exercise class, please post a message on the discussion board on learning central, indicating which parts of the assignment you have been able to solve and what problems you faced. Feedback in the subsequent lecture will address the issues that were highlighted in this way.

Getting Started

In this lab we will continue with the code from the previous lab class. This code is available as `lab1-source.zip` from learningcentral. Please refer to the description of last week's lab class for details on how to compile and run this code.

Implementing search methods

1. Implement the following informed search methods:

   - Uniform-cost search
   - Best-first greedy search

   Note that the only difference with the implementation of A* is in how the nodes in the frontier are ordered. This means that you can essentially keep the implementation from `Astar` and only have to modify the implementation of `AstarNode`.

2. For breadth-first search, A* and each of the implemented methods, answer the following questions:

   (a) Does the method find a solution for the 8-puzzle (within a few seconds)? If not, explain why no solution is found.

   (b) If a solution is found, is it guaranteed to be the optimal one? Explain.

3. Consider the following problem, know as the rickety bridge problem:

   > A number of people need to cross a weak bridge at night. Unfortunately, they have only one torch and the bridge is too dangerous to cross without one. The bridge is only strong enough to support two people at a time. Not all people take the same time to cross the bridge. We assume that there are k people in total, and the time for person p_i to cross the bridge is given by t_i ($i \in \{1, ..., k\}$, $t_i \geq 0$). What is the shortest time needed for all of them to cross the bridge?

   Particular instances of this well-known puzzle often come up during job interviews at companies such as Microsoft and Google. For example, if $k = 4$, and $t_1 = 1$, $t_2 = 2$, $t_3 = 7$ and $t_4 = 10$, the optimal solution is as follows: (i) p_1 and p_2 cross; (ii) p_2 comes back with the torch; (iii) p_3 and p_4 cross; (iv) p_1 comes back with the torch; (v) p_1 and p_2 cross. This gives an optimal total time of $2 + 2 + 10 + 1 + 2 = 17$. We are now interested in solving the general problem (for n people).

Encode this problem as a path searching problem by implementing the classes `WeakBridgeState` and `WeakBridgeAction`, implementing the interfaces State and Action respectively. Use your implementation of A^* search to find optimal solutions. To this end, you will also need to formulate an admissible heuristic.