COMP-424 - Assignment 1

Posted Thursday, January 7, 2010
Due Tuesday, January 19, 2010

Please submit your assignment by e-mail to cs424@cs.mcgill.ca, by 11:59pm

1. [10 points] **Reading**

   As discussed in class, in 1950, Turing published a seminal paper called "Computing Machinery and Intelligence". Read the paper (the pdf is linked from the course schedule page). Summarize in one paragraph the main AI issues that Turing anticipated in this paper.

2. [15 points] **Problem formulation**

   (a) Consider the graph given as an example in Lecture 2 (for tracing search algorithms). Suppose that we want to find a path from the start vertex to the goal vertex such that we always pass through node h. Describe a state-space formulation for this problem. What is the size of the state space in this case?

   (b) Suppose that we have a general graph and we want to find a path from a designated start vertex to a designated goal vertex that passes through k other, given vertices (you want to pick up your friends from their homes before you head off to the party). What is the state space in this case, and what is the size of the state space?

   (c) Suppose that we want to visit *every single vertex* in the graph at least once and end up at the designated goal vertex. What is the state space in this case, and what is the size of the state space?

3. [15 points] **Uninformed search**

   One question that came up in class was that of negative costs. Suppose that we have a problem in which we have negative costs on some edges, but we know a lower-bound, c, on the cost of any edge ($c < 0$). To fix the negative-cost problem, we add $c + 1$ to the costs on all edges, then run uniform-cost search with these new costs. Is this algorithm complete? Is it optimal? Justify your answer *precisely* (i.e. either by a proof or by concrete examples).

4. [15 points]A^* **search**

   As discussed in class, A^* works with heuristics that are admissible (i.e., they under-estimate the true cost of getting to the goal). Under these circumstances, A^* finds the optimal path. But what if the heuristic is not exactly admissible? More precisely, suppose that for any node n, its heuristic value $h(n)$ may overestimate the optimal cost-to-go, but only by a small amount ϵ:

   $$h(n) < h^*(n) + \epsilon, \forall n$$

Let n^* be the optimal goal node, and $g(n^*)$ be the cost of the optimal path. Let n_g be the node returned by running A^* with h as a heuristic. Prove that:

$$g(n_g) \leq g(n^*) + \epsilon$$

Note: You may assume that the function $f = g+h$ is monotonic (so A^* expands nodes in increasing value of f).

5. [30 points] **Problem formulation and heuristics**

   This is problem 3.27 from Russell & Norvig, Third Edition (with a bit different notation). Suppose you have an $n \times n$ grid and n vehicles are on its top row (numbered 1). The vehicles have to get to the bottom row, such that the vehicle that started at index $(1, i)$ must end up at index $(n, n - i + 1)$ (in other words, the order of the vehicle must be reversed). On each time step, each of the vehicles can move one square up, down, left or right, or it can stay put. If the vehicle stays put, one adjacent vehicle (but not more than one) can hop over it. Two vehicles cannot occupy the same square.

   (a) [5 points] Calculate the size of the state space as a function of n

   (b) [5 points] Calculate the branching factor as a function of n

   (c) [5 points] Suppose that vehicle i is at location (x_i, y_i). Write a non-trivial admissible heuristic h_i for the number f moves it will required to get to its goal location, assuming no other vehicles are on the grid.

   (d) [15 points] Which of the following heuristics are admissible for the problem of moving *all* the vehicles to their destination?

      i. $\sum_{i=1}^{n} h_i$

      ii. $\max_{i=1\ldots n} h_i$

      iii. $\min_{i=1\ldots n} h_i$

6. [15 points] **Optimization**

   Suppose you are given a graph $G = (V, E)$ and you want to partition its vertices, into two disjoint sets, V_1 and V_2, such that:

   - The number of vertices in V_1 and V_2 is as close as possible
   - The number of edges where one end is a node in V_1 and end is a node in V_2 is as small as possible.

   (a) Give a function which expresses these optimization criteria

   (b) Explain how you would perform gradient ascent on this function: what are the states, and what is the set of neighbors of a given state?

   (c) Do you expect gradient ascent or simulated annealing to work better on this problem? Why?