Homework 1
Computer Science
Spring 2017
B351

Jonathon Cooke-Akaiwa

January 20, 2017

All the work herein is mine.

Answers

1. (a) we have four directions. This gives up 4 states for our robot. We can travel in a forward direction for any distance, until a wall is encountered. This gives us a single action action. The initial state is at the center of the map. The maze can be considered to be an nxn space. The robot can move to essentially any point on the graph by rotating directions and moving in a forward direction, to any point until a wall is encountered. Our environment will have n^2 locations with 4 possible directions being our states.

   (b) we still have four directions we can orient ourselves towards. Our states states are either at an intersection of two or more corridors, or traveling down a corridor.

   (c) no, we can simply direct the robot to move in one of four directions such as forward, backwards, left, or right rather than telling the robot to orient itself and then move.

   (d) i. In part b. we noted that the only place required to make a turn is at an intersection of two or more corridors. This reduces the number of possible locations from any point down to specific points.

      ii. In part c. we changed the requirement from facing a specific direction down to moving in any of the four directions. This removes the requirement of keeping track of orientation.

      iii. Also from part c. we have stated that movement is now part of changing the robot's direction.

2. def:

3. (a) G(V,E)
   $V = \{U, L, D, K, G, H, J, B, A, F, E, C, I\}$
   $E = \{\{U, L\}, \{U, C\}, \{L, D\}, \{C, I\}, \{D, F\}, \{D, K\}, \{F, E\}, \{K, G\}, \{G, J\}, \{G, H\}, \{J, B\}, \{H, B\}, \{B, A\}\}$

   (b) adj $= \{U : [L, C], L : [U, D], C : [U, I], I : [C], D : [L, F, K], F : [D, E], K : [D, G], E : [F], G : [K, J, H], J : [G, B], H : [G, B], B : [J, H, A], A : [B]\}$

   (c) A trace of a DFS of G could be: UCILDFEKGJBAH

   (d) We have 12 rooms that need to be scanned. At 4 minutes per room we have a fixed cost of 48 minutes that cannot be eliminated. This fixed cost assumes that a room does not need to be scanned more than once. With the exception of the transition between rooms L and C, the move cost between any two rooms is 2 minutes. The cost for the aformentioned transition from L to C is 7. With this considered we need to then follow a path and calcuate an approximate amount if time taken to check the floor. Let us start in room I. Following a DFS gives us the following transitions with their cumulative cost:

Transition	Cost	total time elapsed
$I \rightarrow C$	2	2
$C \rightarrow L$	7	9
$D \rightarrow F$	2	11
$F \rightarrow E$	2	13
$E \rightarrow F$	2	15
$F \rightarrow D$	2	17
$D \rightarrow K$	2	19
$K \rightarrow G$	2	21
$G \rightarrow J$	2	23
$J \rightarrow B$	2	25
$B \rightarrow A$	2	27
$A \rightarrow B$	2	29
$B \rightarrow H$	2	31

Taking our cost of movement as 31 minutes and adding that to our fixed cost of scanning new rooms of 48 minutes gives us a total time of 79 minutes.
To annotate the graph to provide this information we would need to add a weight or cost to each transition that accounts for the time to move between rooms. This weight could be used in addition to cost of scanning unique rooms to produce a total cost, similar to the steps outlined above. Each room that was visited could be added to a set and the fixed cost calcuated from there. When considering the transition between L and C via U we could replace this with an edge with a cost of 7 minutes. This might be a useful change to make as we would not have to exclude node U from our calculation of room scans while also simplifying the calcuation of transitions.

(e) Based on my estimate in the previous question it would take two batteries to fully scan the floor. This assumes there is no travel cost associated with changing batteries; i.e. if the drone must return home to charge.

(f) see p1.py

(g)

(h)

4. (a)

Game No	Human Win%	Comp Win%
H1	40%	40%
H2	20%	40%
H3	40%	10%
H4	60%	0%
H5	60%	20%

These game results give us an expectation of 44% for a human winning. This did seem much lower than I expected, but I did not consider the ability to tie a game. After considering the ability for two players to tie during a game of rock paper sissors gives us a human win rate that is higher than what should have been expected. The most interesting stat was actually that the computer only beat the human in win rate once, in every other case it did equally well or worse. The computers expected win rate is only 22%.

(b) The python library random will uniformly pick an integer using its pseudo-random generator. While it is not a truely random number that is generated, there is no bias when generating a number. In my case I tried a number of options including picking the same number, rotating through possible numbers, and also trying to pick a random number each time. Each of these methods I used are based on the previous number selected, even my 'random' selection.

(c) see p2.py