Homework 1
Computer Science
Spring 2017
B351

Steven Myers

January 17, 2017

All the work herein is mine.

Answers

1. (a) The state space is as large as there are valid "moves" that the robot can make. We can consider every location within the maze as a "state." We can make some assummptions, such as the maze consists of only rectangular corridors and that the robot moves roughly from square to square within this maze. We could then take every tile in the maze that is not blocked (a wall), and sum the number of possible moves on unblocked tiles that would not result the robot moving into a blocked tile. If necessary, we must also dictate what orientation, and moreover how much distance can be covered from each state as well.

   (b) If we only move until we reach intersections of corridors, it means that we can join long stretches of tiles in which there are no intersections and treat it as one state, since we are no longer interested in states that do not break a cardinal direction (e.g something that only travels north).

   (c) This does little to change the overall problem of exiting the maze with our robot. The orientation of the robot is identical to the last action taken. So, if the last time we had a transition from one state to another, our action was to move east, then the Robot's orientation would also be east.

   (d) We made the following simplifications:
      - A robot cannot stop moving until it reaches an intersection.
      - We do not know the size of maze or whether or not the robot can successfully navigate itself out.
      - We do not know the shape of the maze, but assume it consists of rectangular walls.

2. A problem in which an AI is not well suited for could be any of the following:
   - Determining the best tasting wine in a wine-tasting contest
   - Picking out the prettiest dress in a wardrobe
   - Determining whose life is more valuable than another (Will Smith's family in *iRobot*)

3. (a) Here is a representation of the floor in a graph $G = \{V, E\}$:

```
G = {V, E}
V : {U, A, B, H, C, I, J, G, F, D, K, E, L}
E : {(U, C), (U, L), (A, B), (B, H), (B, J), (H, G), (C, I), (J, G), (G, K),
    (F, E), (F, D), (D, K), (D, L)}
```

(b) Here is an adjacency list representation of the floor:

```
U => {C, L}
A => {B}
B => {A, H, J}
H => {B, G}
C => {U, I}
I => {C}
J => {B, G}
G => {H, J, K}
F => {E, D}
D => {F, K, L}
K => {G, D}
E => {F}
L => {U, D}
```

(c) Here is one possible sequence of a DFS search on the floor:
U, C, I, L, D, F, E, K, G, J, B, A, H

(d) Since we know that it takes 4 minutes to scan every room, we can multiply the number of minutes times the number of vertices to find the total amount of time spent scanning as a constant value. So, it will take 48 minutes to scan the entire floor since there are 12 rooms. Since there is only one path to travel from U to I, we can optimize the path by beginning our drone in room I. This also guarantees that we don't have to make a roundtrip between C & L.

4.