Portland State UNIVERSITY

CS 441/541: Artificial Intelligence, Winter 2023

Homework #2

Note: This assignment is due **Wednesday, 2/15 @ 10pm.** Please email your typed or written (scanned) homework solutions to our TA by the due date. Write and exposit clearly – show a sufficient amount of work on each problem, without over indulging, please. Text nomenclature refers to 3rd edition of R&N.

Each student will turn in an individual assignment (so that we have something upon which to base your individual grade). However, you are encouraged to discuss and work through these problems with your instructor, TA and above all, other students in our class. B**ottom line**: the answers and work you submit will authentically be your own.

1. Prove/argue the following:

   (i) The maximum number of leaves in a binary tree of depth *d* is 2^d.

   (ii) The maximum number of nodes in a binary tree of depth *d* is 2^{d+1}-1.

2. Exercise 3.2 from the text.

3.2 Your goal is to navigate a robot out of a maze. The robot starts in the center of the maze

facing north. You can turn the robot to face north, east, south, or west. You can direct the robot to move forward a certain distance, although it will stop before hitting a wall.

a. Formulate this problem. How large is the state space?

b. In navigating a maze, the only place we need to turn is at the intersection of two or more corridors. Reformulate this problem using this observation. How large is the state space now?

c. From each point in the maze, we can move in any of the four directions until we reach a turning point, and this is the only action we need to do. Reformulate the problem using these actions. Do we need to keep track of the robot's orientation now?

d. In our initial description of the problem we already abstracted from the real world, restricting actions and removing details. List three such simplifications we made.

3. Exercise 3.3.

3.3 Suppose two friends live in different cities on a map, such as the Romania map shown in Figure 3.2. On every turn, we can simultaneously move each friend to a neighboring city on the map. The amount of time needed to move from city i to neighbor j is equal to the road distance $d(i, j)$ between the cities, but on each turn the friend that arrives first must wait until the other one arrives (and calls the first on his/her cell phone) before the next turn can begin. We want the two friends to meet as quickly as possible.

a. Write a detailed formulation for this search problem. (You will find it helpful to define some formal notation here.)

b. Let $D(i, j)$ be the straight-line distance between cities i and j. Which of the following heuristic functions are admissible? (i) $D(i, j)$; (ii) $2 \cdot D(i, j)$; (iii) $D(i, j)/2$.

c. Are there completely connected maps for which no solution exists?

d. Are there maps in which all solutions require one friend to visit the same city twice?

4. Exercise 3.4.

3.4 Show that the 8-puzzle states are divided into two disjoint sets, such that any state is reachable from any other state in the same set, while no state is reachable from any state in the other set. (*Hint:* See Berlekamp *et al.* (1982).) Devise a procedure to decide which set a given state is in, and explain why this is useful for generating random states.

5. Exercise 3.14.

3.14 Which of the following are true and which are false? Explain your answers.

a. Depth-first search always expands at least as many nodes as A* search with an admissible heuristic.

b. $h(n) = 0$ is an admissible heuristic for the 8-puzzle.

c. A* is of no use in robotics because percepts, states, and actions are continuous.

d. Breadth-first search is complete even if zero step costs are allowed.

e. Assume that a rook can move on a chessboard any number of squares in a straight line, vertically or horizontally, but cannot jump over other pieces. Manhattan distance is an admissible heuristic for the problem of moving the rook from square A to square B in the smallest number of moves.

6. Exercise 3.18.

3.18 Describe a state space in which iterative deepening search performs much worse than depth-first search (for example, $O(n^2)$ vs. $O(n)$).

7. Exercise 3.25.

3.25 The **heuristic path algorithm** (Pohl, 1977) is a best-first search in which the evaluation function is $f(n) = (2 - w)g(n) + wh(n)$. For what values of w is this complete? For what values is it optimal, assuming that h is admissible? What kind of search does this perform for $w = 0$, $w = 1$, and $w = 2$?

8. Exercise 3.26.

3.26 Consider the unbounded version of the regular 2D grid shown in Figure 3.9. The start state is at the origin, (0,0), and the goal state is at (x, y).

a. What is the branching factor b in this state space?
b. How many distinct states are there at depth k (for $k > 0$)?
c. What is the maximum number of nodes expanded by breadth-first tree search?
d. What is the maximum number of nodes expanded by breadth-first graph search?
e. Is $h = |u - x| + |v - y|$ an admissible heuristic for a state at (u, v)? Explain.
f. How many nodes are expanded by A* graph search using h?
g. Does h remain admissible if some links are removed?
h. Does h remain admissible if some links are added between nonadjacent states?

9. Exercise 4.1.

4.1 Give the name of the algorithm that results from each of the following special cases:

a. Local beam search with $k = 1$.
b. Local beam search with one initial state and no limit on the number of states retained.
c. Simulated annealing with $T = 0$ at all times (and omitting the termination test).
d. Simulated annealing with $T = \infty$ at all times.
e. Genetic algorithm with population size $N = 1$.

10. Exercise 4.5.

4.5 The AND-OR-GRAPH-SEARCH algorithm in Figure 4.11 checks for repeated states only on the path from the root to the current state. Suppose that, in addition, the algorithm were to store *every* visited state and check against that list. (See BREADTH-FIRST-SEARCH in Figure 3.11 for an example.) Determine the information that should be stored and how the algorithm should use that information when a repeated state is found. (*Hint*: You will need to distinguish at least between states for which a successful subplan was constructed previously and states for which no subplan could be found.) Explain how to use labels, as defined in Section 4.3.3, to avoid having multiple copies of subplans.

11. Exercise 4.10.

4.10 Consider the sensorless version of the erratic vacuum world. Draw the belief-state space reachable from the initial belief state $\{1, 2, 3, 4, 5, 6, 7, 8\}$, and explain why the problem is unsolvable.

12. Exercise 4.14.

4.14 Like DFS, online DFS is incomplete for reversible state spaces with infinite paths. For example, suppose that states are points on the infinite two-dimensional grid and actions are unit vectors $(1, 0)$, $(0, 1)$, $(-1, 0)$, $(0, -1)$, tried in that order. Show that online DFS starting at $(0, 0)$ will not reach $(1, -1)$. Suppose the agent can observe, in addition to its current state, all successor states and the actions that would lead to them. Write an algorithm that is complete even for bidirected state spaces with infinite paths. What states does it visit in reaching $(1, -1)$?

13. Write a short program to execute the Gradient Descent (GD) algorithm as described in class. Recall that the key steps in GD are as follows:

$$\mathbf{x}_0 = random$$
$$\mathbf{x}_t = \mathbf{x}_{t-1} - \eta \nabla f(\mathbf{x}_{t-1})$$

Apply GD to approximately solve for the global minimum of the function $f(x, y) = 5x^2 + 40x + y^2 - 12y + 127$.

You will run (3) sets of experiments using different values for η: (i) η = .1, (ii) η =.01, and (iii) η =.001. Run GD for 500 steps for each experiment; in each case initialize $x_0 \in [-10,10] \times [-10,10]$.
R*eport the best performance out of* 10 trials for each of the different η value cases. Provide some comments and analysis about your results.

Please include your (concise) GD code in your assignment write-up.