CS 124 Data Structures and Algorithms — Spring 2018

Problem Set 2

Due: 11:59pm, Wednesday, February 14th

See homework submission instructions at
`http://sites.fas.harvard.edu/~cs124/cs124/problem_sets.html`

Problem 4 is worth 40% of this problem set, and problems 1-3 constitute the remaining 60%.

For each problem where you are asked to give an algorithm, more points are given for asymptotically faster algorithms. In judging the number of points to award a correct solution, we only consider the running time in asymptotic notation, i.e. a writeup of an algorithm taking $1000n^2$ steps versus one taking $.01n^2$ steps would receive the same number of points — both would be simply treated as $\Theta(n^2)$-time solutions.

1 Problem 1

A min-ordered binary heap stores keyed items in a rooted tree in which each node has two children and all leaves are all roughly at the same depth (if n, the number of items being stored, is not of the form $2^k - 1$, then some leaves will be one level shallower than othres). The min-ordered property says that if u is a parent of v, then the key of the item in node u is at most that of node v.

Suppose we instead use a min-ordered b-ary heap, which is similar to the above except that each internal node in the tree now has b children instead of 2 (except for possibly one internal node, since n may not be of the form $1 + b + b^2 + \ldots + b^k$ for some k.

(a) (7 points) Describe how you would modify the implementation of DecreaseKey, Insert, and DeleteMin from binary heaps to work for b-ary heaps.

(b) (3 points) How would you choose b to optimize the running time of Dijkstra's algorithm when using a b-ary heap (getting the right answer up to a constant factor is sufficient)?

2 Problem 2

You are given an $n \times n$ pixelated image, where each pixel is 1mm × 1mm. Some pixels are black and others are white. White pixels should be considered as empty. If two black pixels are adjacent (where each pixel p has eight adjacent pixels, i.e. above, below, to the right, to the left, and diagonal from it in all four diagonal directions), then we say those two pixels are part of the same **object**. Being part of the same object is transitive: if black pixels a and b are part of the same object and b and c are, then a and c are as well. The **perimeter**

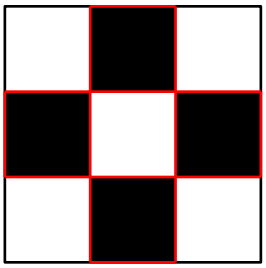

Figure 1: Example pixelated image containing only one object, with perimeter 16. The edges on the perimeter are highlighted in red.

of an object is the sum of side lengths of pixels in that object such that *either* the pixel side is on the boundary of the image, *or* on one side a black pixel, and on the other side is a white pixel.

Describe an algorithm which, given the description of a pixelated image stating which pixels are black and white, outputs the number of objects, and for each object outputs its perimeter. **Hint:** to identify the objects in the image, construct an appropriate graph based on the input image and run one of the algorithms from class on it.

3 Problem 3

You are Blink, the protagonist in the game Legend of Griselda. You live on an $M \times N$ two-dimensional grid, where each grid cell is labeled with a pair $(x, y) \in \{1, \ldots, M\} \times \{1, \ldots, N\}$. You are trying to reach position (M, N), where a certain dragon lives that you must defeat. Your weapon of choice is a bow and arrow, and you plan to shoot the dragon upon arrival in his grid cell. You start at cell $(1, 1)$. For each of the problem parts below, when you are asked to give an algorithm, you should also analyze its running time.

(a) (2 points) Suppose that some grid cells are open fields you can walk through, but others are mountainous terrain that are impassable. The cells $(1, 1)$ and (M, N) are open fields. At each step you can walk in one of the four cardinal directions: up, down, left, and right, though not into impassable cells. Describe an algorithm to find the shortest time to reach the dragon. If impossible, your algorithm should say so.

(b) (3 points) You need to eat, and suppose that some cells with open fields have edible plants to sustain you, but others don't. You start the game with a "life counter" of 5 lives, and every time you walk through an open field that happens to be a desert, you go hungry a bit and lose some of your vitality, causing your life counter to decrease by 1. (There's nothing you can ever do later to make it increase again.) You need to make sure that you have at least 1 life counter when reaching the dragon. Now describe an algorithm to reach the dragon in the shortest time with at least 1 life counter left. If impossible, your algorithm should say so. You are promised that neither cell $(1, 1)$ nor

cell (M, N) is a desert. **Hint:** how can you modify your world representation from part (a) to take into consideration your five lives?

4 Programming Problem

Solve ESCAPE on the programming server (`https://cs124.seas.harvard.edu`).