University of Waterloo
CS240 Fall 2017
Assignment 4

Due Date: Wednesday, July 6, at 5:00pm

Please read `http://www.student.cs.uwaterloo.ca/~cs240/s16/guidelines.pdf` for guidelines on submission. This assignment contains both written problems and a programming problem. Submit your written solutions electronically as a PDF with file name a04wp.pdf using MarkUs. We will also accept individual question files named a04q1w.pdf, a04q2w.pdf, ... , a04q5w.pdf if you wish to submit questions as you complete them.

Problem 5 contains a programming question; submit your solution electronically as a file named `kdpartition.cpp`.

Problem 1

c) Idea: the idea here is to simultaneously traverse and compare the input quadtrees and build a new one. The traversal is as you would expect, recursively search each of the 4 children in order. If one quadtree reaches a leaf node before the other, continue searching down the quadtree that has more children and compare (intersect or union) with the value at which the first quad tree stopped.

Problem 2

b) (6,2) (4,6) (2,9) (1,7)

Problem 3

b) Modify the 1-dimensional range trees to include, at each node, not just the point at the node but the number of nodes in the subtree rooted at that node. Let "boundary nodes", "inside nodes" and "outside nodes" be as described in the course slides. To determine the number of points in a range $[k_1, k_2]$, add up the number of boundary points that are in range, together with the sum of the sizes of the subtrees rooted at the "top" internal nodes.

Note: only check the root of the inside subtrees.

Checking boundary points is $O(\log n)$ time and since inside subtrees can only occur as the right child of a node on the left boundary or the left child of a node on the right boundary, there are $O(\log n)$ inside subtrees to check. Counting the size of the subtree at one node is $O(1)$ time so overall the run-time will be $O(\log n)$.

c) Build a 2-dimensional range tree as described in the course slides, but use the modified tree from part (a) for the associated trees on the y-coordinates. To do a range counting

query, modify the recipe from course slides as follows. For every "top" inside node v, perform a range counting query on the associated tree on the y-coordinate. Return the sum of the results of all the range counting queries on the associated trees, together with the number of boundary nodes that are within the region R.

Problem 4

a) For each of the following pattern strings, determine the Knuth-Morris-Pratt failure array:

i) `F[j] 0 0 0 1 0 0 0 0 1 0 1 0 0 1 2 3 0 0 1`

ii) `F[j] 0 0 0 1 0 1 0 1 2 3 4 5 6 7 8 9 10 11`

iii) `F[j] 0 0 0 1 0 1 0 1 2 3 4 5 6 0 1 2 3 4`

iv) `F[j] 0 0 0 1 0 1 0 1 2 3 4 5 6 0 1 2 3 4`

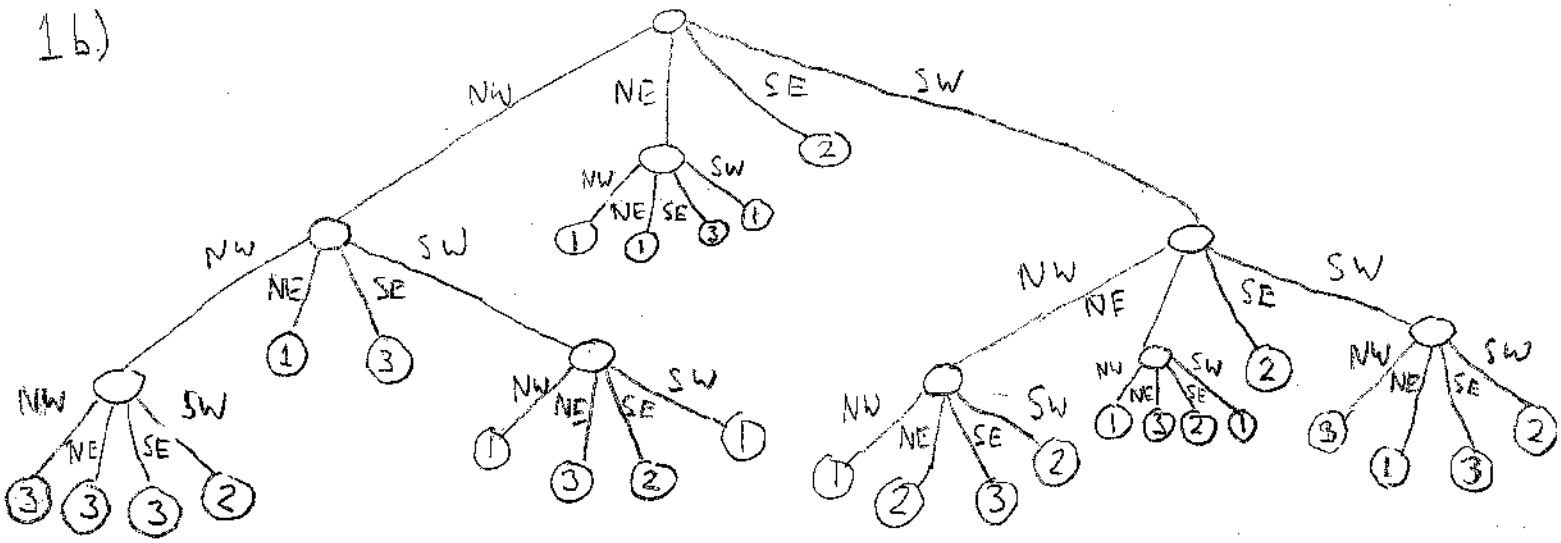
b) The idea for this question is that depending on the value of text matched with `?`, there can be different results for the pattern to jump ahead. Ideally, you can make a multidimension failure array to consider the cases where `?` is a character that already appears in the pattern and also the case where it matches a character not in the pattern. This can then be reduces to combine cases together if we wish to reduce space.

The KMP algorithm is then altered by not only using the index $j - 1$ (the number of characters matched in the pattern and the text but with added parameter c the character from the text that was matched up with `?`.

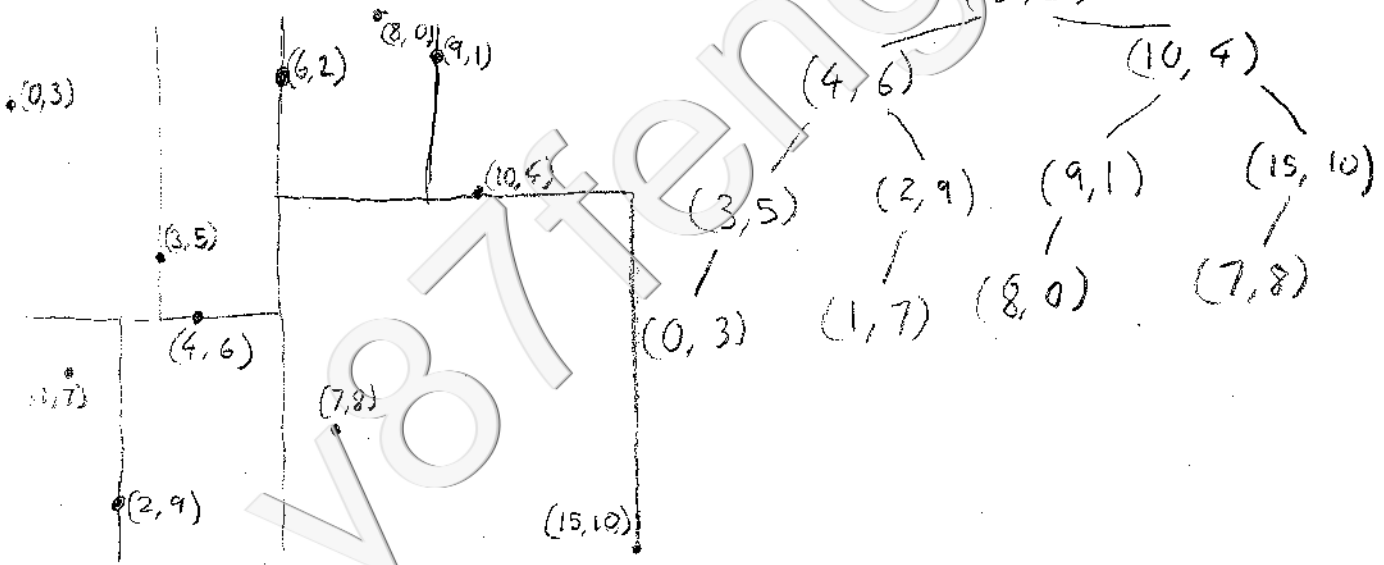
Problem 5

b) The important part of this question is that you should not be sorting points at each step. Instead, sort the points once by x-coordinate and once by y-coordinate. When you subdivide a section of points, you are simply going through the initially sorted coordinate lists and pulling out points that are in the region. You can go through the points in linear time and after picking them out, they are already in sorted order.

1b)



2a)



2b) Order examined:

(6,2) (4,6) (3,5) (0,3) (2,9) (1,7) (10,4) (9,1) (8,0) (15,10) (7,8)

Results:

(6,2) (4,6) (3,5)

3a)