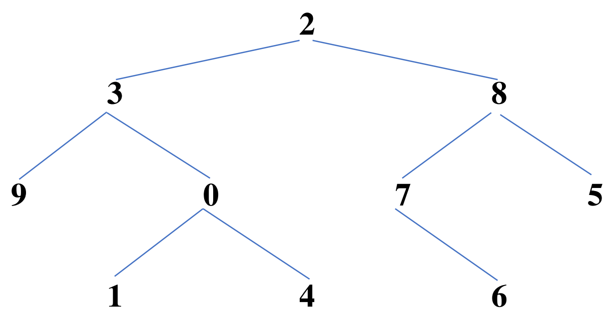
CS 452 - Design and Analysis of Algorithms

Homework 5

John Unger

Question 1: Tree traversals

a. Draw a binary tree with ten nodes labeled 0, 1, 2, ..., 9 in such a way that the inorder and postorder traversals of the tree yield the following lists: 9, 3, 1, 0, 4, 2, 7, 6, 8, 5 (inorder) and 9, 1, 4, 0, 3, 6, 7, 5, 8, 2 (postorder).



b. Give an example of two permutations of the same n labels 0, 1, 2, ..., n−1 that cannot be inorder and postorder traversal lists of the same binary tree. This means that you need to find an inorder traversal list A of a tree, which cannot have the postorder traversal list B. You may pick a specific number n ≥ 10 for your example.

A tree with n = 12

Inorder: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}

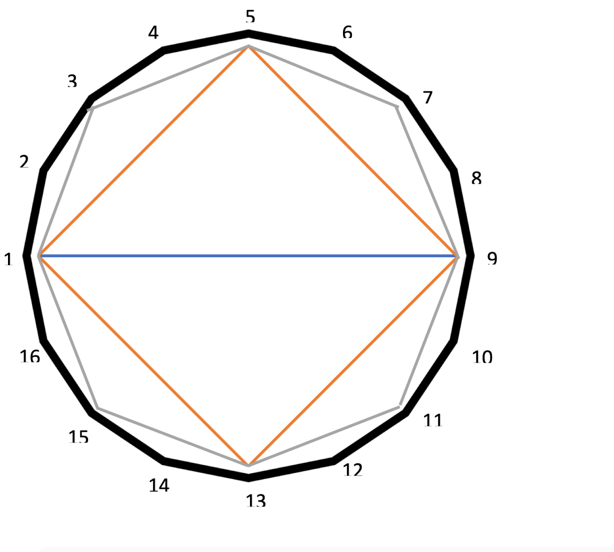
Postorder: {2, 3, 5, 6, 8, 9, 11, 12, 4, 7, 10, 1}

(The lists could be identical, since the structure is different, the trees would still differ.)

Question 2: Quickhull worst case

Give a concrete example of the worst case for the convex hull problem using the Quickhull algorithm for n=16 points in the plane. Demonstrate how your example is a worst case for the algorithm.

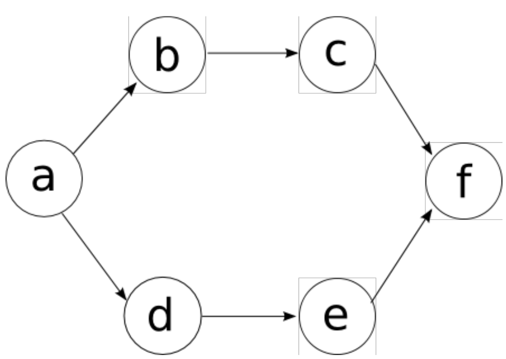
The Quickhull algorithm works like quicksort. It connects the extremes, and then adjusts the polygon to include the points not already contained by each subsection. The worst-case scenario, with 16 points on the plane, would have the points all part of the boundaries, while not already being contained by a pervious line. Essentially making all 16 points would be vertexes of a circular shape.



As this illustration is showing, a worst case would have 16 points set up to be in a circular shape. The Quickhull algorithm would first connect points 1 & 9, since they are the extremes. No other points would be eliminated that would need computation. Then the algorithm would incorporate points 5 & 13, since they are the furthest away from the original segment. Still no other points would be eliminated. And so, the algorithm would continue, having to compute each individual point since it would not be incorporated in the convex polygon otherwise.

Question 3: Topological ordering

Consider the directed acyclic graph below. How many topological orderings does it have? List all of them.



1. a b c d e f

2. a b d c e f

3. a b d e c f

4. a d e b c f

5. a d b c e f

6. a d b e c f

There are 6 permutations to these directed orderings.

Question 4: Minimum and maximum value

One can easily find both the minimum and maximum value of a set of numbers by iterating twice through the set and making n-1 comparisons each time for a total of 2n-2 comparisons. Describe a way to simultaneously find the minimum and maximum value from a set of numbers that uses only 3*ceiling(n/2) comparisons.

```
Pair MaxMin(array, array_size)
   if array_size = 1
      return element as both max and min
   else if arry_size = 2
      one comparison to determine max and min
      return that pair
   else // array_size > 2
      recur for max and min of left half
      recur for max and min of right half
      two comparisons determine the true max and min of the two candidates
      return the pair of max and min
```

By dividing and conquering the array, we are able to cut the number of comparisons. Just like the quicksort, we divide the array in two and compare the local max and min recursively. In case there is only 1 item in the array, the max and min are the same value. After all recursive calls for each half, we make one iteration for the maximum that is true over the whole array, and the same for the minimum.