CS 577 **Assignment 6 – Divide and Conquer 2** Fall 2022

Answer the questions in the boxes provided on the question sheets. If you run out of room for an answer, add a page to the end of the document.

Name: Ramapriya Ranganath Wisc id: rranganath

Divide and Conquer

1. *Kleinberg, Jon. Algorithm Design (p. 248, q. 5)* Hidden surface removal is a problem in computer graphics where you identify objects that are completely hidden behind other objects, so that your renderer can skip over them. This is a common graphical optimization.

   In a clean geometric version of the problem, you are given n non-vertical, infinitely long lines in a plane labeled $L_1 \dots L_n$. You may assume that no three lines ever meet at the same point. We call L_i "uppermost" at a given x coordinate x_0 if its y coordinate at x_0 is greater than that of all other lines. We call L_i "visible" if it is uppermost for at least one x coordinate.

   Give an algorithm that takes n lines as input and in $O(n \log n)$ time returns all the ones that are visible. (See the figure for an example.)

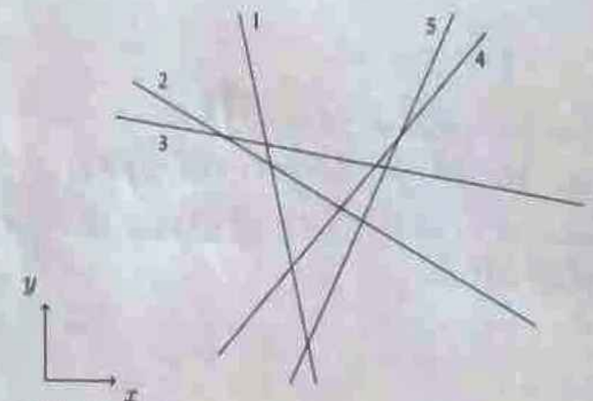

Figure 5.10 An instance of hidden surface removal with five lines (labeled *1-5* in the figure). All the lines except for *2* are visible.

Solution: Since that no 3 lines ever meet at the same point we sort lines by slope and add them one by one.
Input :- n lines Output :- solutions.
sort n line based on slope, we have $l_1 \dots l_n$. Let $S = \{l_1, l_2\}$
Let I_1, I_2 denote the most recent added lines in S when adding a new line. for i = 3 ... n, we compute interest of I_1 I_2 · $A(x_A, y_A)$ onwards : $L_i = ax + by + c = 0$, while $ax_A + by_A + c < 0$
y_A smaller than the y point on l_i with same x_A value. remove I_1 from S get new I_1, I_2 get new $A(x_A, y_A)$ add l_i to S. Sorting is $O(n \log n)$ if we do not enter while loop, which means we do not remove lines we just add lines one-by-one this is $O(n)$. If we enter while loop since each line can only be remove once, we have $O(n)$ ∴ It is $O(n \log n)$

2. In class, we considered a divide and conquer algorithm for finding the closest pair of points in a plane. Recall that this algorithm runs in $O(n \log n)$ time. Let's consider two variations on this problem:

   (a) First consider the problem of searching for the closest pair of points in 3-dimensional space. Show how you could extend the single plane closest pairs algorithm to find closest pairs in 3D space. Your solution should still achieve $O(n \log n)$ run time.

   Solution: We have P_x, P_y, P_z as sorted x, y, z values $O(n\log n)$. [illegible] a hyperplane $x = t$ divides space into Q (left half of P_x) and R (right half of P_x) then separates corresponding y, z values to assigned points in Q, R ($O(n)$) divides based on y, z and do this recursively. We have min $\{d(q_1^2, q_1^2), d(r_2^2, r_1^2)\}$. Then we use lemma in lecture, divide portion around hyperplane check points ($O(n)$) then total complexity $O(n\log n)$

   (b) Now consider the problem of searching for the closest pair of points on the surface of a sphere (distances measured by the shortest path across the surface). Explain how your algorithm from part a can be used to find the closest pair of points on the sphere as well.

   Solution: We use the same algorithm, we consider points on sphere as on subset of larger 3D space. We modify our distance metric $d(x_0, x_1)$ to distance on sphere. Then we use the same algorithm

   (c) Finally, one way to approximate the surface of a sphere is to take a plane and "wrap" at the edges, so a point at x coordinate 0 and y coordinate MAX is the same as x coordinate 0 and y coordinate MIN. Similarly, the left and right edges of the plane wrap around. Show how you could extend the single plane closest pairs algorithm to find closest pairs in this space.

   Solution: We treat surface on sphere as a plane. Then we implement regular closest pair algorithm on this single plane. Then for checking the points in δ subset of points within δ of L, we also add points within δ of edge of plane into set δ. We padding the edge use the otherside of points we copy the points at lower δ-edge of plane and add to upper of plane. Then we not only check pairs cross divides, check pairs across edge. Then the following algorithm is the same.

3. *Erickson, Jeff. Algorithms (p. 58, q. 25 d and e)* Prove that the following algorithm computes $\gcd(x, y)$ the greatest common divisor of x and y, and show its worst-case running time.

```
BINARYGCD(x,y):
if x = y:
    return x
else if x and y are both even:
    return 2*BINARYGCD(x/2,y/2)
else if x is even:
    return BINARYGCD(x/2,y)
else if y is even:
    return BINARYGCD(x,y/2)
else if x > y:
    return BINARYGCD( (x-y)/2,y )
else
    return BINARYGCD( x, (y-x)/2 )
```

1) Using Invariant. Consider k is the GCD of x and y). x = y then return y = x.

•) x & y both are even, then k is even and k is of x/2, y/2.

•) x is even y is odd, then k is odd theory x, y is divisible by x/2, y.

•) x is odd and y is even, same result

•) ~~x ...~~ •) x & y are odd, $d = \frac{x-y}{2}$ and vice versa.

Solution:

•) Then we show correction. Since the algorithm returns the correct result, and algorithm terminates in $\log[x]+1 + \log[y]+1$. Then the algorithm is correct

•) Worst case:- Then recursion will stop when x = y and when x = y = 1 ∴ The running time is $O(\log x + \log y)$

4. Here we explore the structure of some different recursion trees than the previous homework.

   (a) Asymptotically solve the following recurrence for $A(n)$ for $n \geq 1$.

$$A(n) = A(n/6) + 1 \qquad \text{with base case} \qquad A(1) = 1$$

Solution:

n
↓
$n/6$ 1
↓
$n/36 = \frac{n}{6^2}$ 1
↓
$\frac{n}{6^3}$ 1
⋮
↓
$\frac{n}{6^k}$

$A(n) \leq \sum_{i=0}^{k} 1 = k$

$k = \log_6 n$

$O(\log n)$

(b) Asymptotically solve the following recurrence for $B(n)$ for $n \geq 1$.

$$B(n) = B(n/6) + n \quad \text{with base case} \quad B(1) = 1$$

Solution:

$B(n) = B\left(\frac{n}{6}\right) + n$

$= B\left(\frac{n}{36} + n\right) + n = B\left(\frac{n}{6^2}\right) + 2n$

$\vdots$

$B\left(\frac{n}{6^k}\right) + kn$

$B(n) = n \cdot \sum_{i=0}^{k} \frac{1}{6^i}$

$n \cdot \frac{1 - \left(\frac{1}{6}\right)^k}{1 - \frac{1}{6}}$

$\therefore O(n)$

(c) Asymptotically solve the following recurrence for $C(n)$ for $n \geq 0$.

$$C(n) = C(n/6) + C(3n/5) + n \quad \text{with base case} \quad C(0) = 0$$

Solution:

n n

$n/6$ $\frac{3n}{5}$ $\frac{23}{30} n$

$\frac{n}{36}$ $\frac{3n}{30}$ $\frac{3n}{30}$ $\frac{9n}{25}$ $\left(\frac{23}{30}\right)^2 n$

$C(n) \leq n \sum_{i=0}^{k} \left(\frac{23}{30}\right)^i$

$\leq n \frac{1 - \left(\frac{23}{30}\right)^k}{1 - \left(\frac{23}{30}\right)}$

$O(n)$

(d) Let $d > 3$ be some arbitrary constant. Then solve the following recurrence for $D(x)$ where $x \geq 0$.

$$D(x) = D\left(\frac{x}{d}\right) + D\left(\frac{(d-2)x}{d}\right) + x \quad \text{with base case} \quad D(0) = 0$$

Solution:

x

$\frac{x}{d}$ $\frac{x(d-2)}{d}$ $\frac{d-1}{d} x$

$\frac{x}{d^2}$ $\frac{(d-2)x}{d^2}$ $\frac{x(d-2)}{d^2}$ $\left(\frac{(d-2)x}{d}\right)^2$ $\left(\frac{d-1}{d}\right)^2 x$

$x \cdot \frac{1 - \left(\frac{d-1}{d}\right)^k}{1-d}$

$O(x)$