Analysis of Algorithms II

Project 2

Report

Ahmet Göktuğ SEVİNÇ

150140120

21.04.2018

1)

Master Theorem is a method that is used for asympthotic analysis of divide and conquer recurrences. The general formula of recurrence relation is:

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

Here ‘a’ shows how many sub-problems we are dividing the problem, and ‘n/b’ show the size of each sub-problem.

2)

In the project we are asked to find the closest pair of points in 3D space. For this purpose, I maintained two lists Px and Py, each containing all points in a sorted order of x-axis. Then, by calling recursive function that regenerates Px and Py by simply dividing them to two parts such as LeftX, RightX, LeftY, RightY and searching closest pair of points in Left X-Y and Right X-Y. If the size of these arrays becomes less than or equal to 3, I applied brute force to find the minimum distance between these points and then, simply found the smallest result of Left and Right arrays. Here, the problem is, I am finding the minimum distances only within Px or Py but there might be a smaller distance between the points of Px and Py. If there is such a pair of points between Px and Py, these point has to be close to each other where Px and Py separates from each other. Since Px and Py are separated from one another with respect to x-axis, they are separated with a plane in 3D, that is perpendicular to x-axis and parallel to y and z axis, such as ‘x = a’ plane. Now, lets say smallest distance that we have found before -smallest distance of Left array or Right array- is ‘d’. If there is a pair of points between Left and Right arrays that has a distance smaller ‘d’, their difference between x coordinates can be at most ‘d’ to satisfy the condition. So, I maintained an array S, that collects these points. And after each Left and Right distance calculations, I also calculated the distances between the points in S array, and assigned minimum to smallest distance of Left, Right or S arrays. To find the minimum distance of points of S array, first I sorted its points with respect to y-axis. Since we know their distance between x coordinates is at most ‘d’, their distance of z and y pair has to be smaller than ‘d’, so I checked this condition at inner loop and found if there is a pair of points that has a distance smaller than ‘d’.

3)

```
ClosestPair(P, int n)
   Constract Px and Py                                    O(nlogn)
   Sort Px and Py
   float smallest = ClosestPair_Rec(Px, Py, n)

ClosestPair_Rec(Px, Py, n)
   If n<=3 then find the closest pair by measuring all Euclidean
distances.
   Endif

   mid = n/2
   midPoint = Px[mid]

   Construct Lx, Rx, Ly, Ry                                 O(n)
   float dl = ClosestPair_Rec(Lx, Ly, mid)                 2T(n/2)
   float dr = ClosestPair_Rec(Rx, Ry, n-mid)

   d = min(dl, dr)
   S: Points in Px within distance d from midPoint          O(n)
   return min(d, sClosest(S, j, d))

sClosest(S, j, d)
   Construct Sy;                                            O(n)

   For each point s in Sy, find the distances from s to     O(n)
points that has a distance between y-z coordinates smaller than d.
   If there is such s,s' pair of points, calculate Euclidean
distance and if it is smaller than d assign result to minimum.
```

Sorting S array would take O(nlogn) time but there are several points in a distance of d from mid-plane. So we can consider it as O(n). And the inner loop that finds distances of points in S will not execute more than a constant number so we can take it also as O(n).

$T(n) = 2T(n/2) + O(nlogn) + 4O(n)$

$T(n) = 2T(n/2) + O(nlogn)$

$T(n) = O(n \times logn \times logn)$

4)

	1000	5000	10000	25000
Elapsed Time	0.002sec	0.014sec	0.023	0.064
Total comparison	1080	6025	11898	32286
n x logn x logn	99.201	753.992	1.763.584	5.277.908

(Elapsed Time) 5000 = 7x(Elapsed Time) 1000 → (nlognlogn)5000 = 7x(nlognlogn)1000

(Elapsed Time) 10000 = 2,3x(Elapsed Time) 5000 → (nlognlogn)10000 = 2,3x(nlognlogn)5000

(Elapsed Time) 10000 = 2,8x(Elapsed Time) 25000 →(nlognlogn)25000 = 2,9x(nlognlogn)10000

Also total number of comparisons has similar relationship with each other.