Problem Set 1

This problem set is due **at 11:59pm** on **Thursday, February 12, 2015.**

This assignment, like later assignments, consists of *exercises* and *problems*. Hand in solutions to the problems only. However, we strongly advise that you work out the exercises also, since they will help you learn the course material. You are responsible for the material they cover.

Please turn in each problem solution separately. Each submitted solution should start with your name, the course number, the problem number, your recitation section, the date, and the names of any students with whom you collaborated.

We will often ask you to "give an algorithm" to solve a problem. Your write-up should take the form of a short essay. Start by defining the problem you are solving and stating what your results are. Then provide:

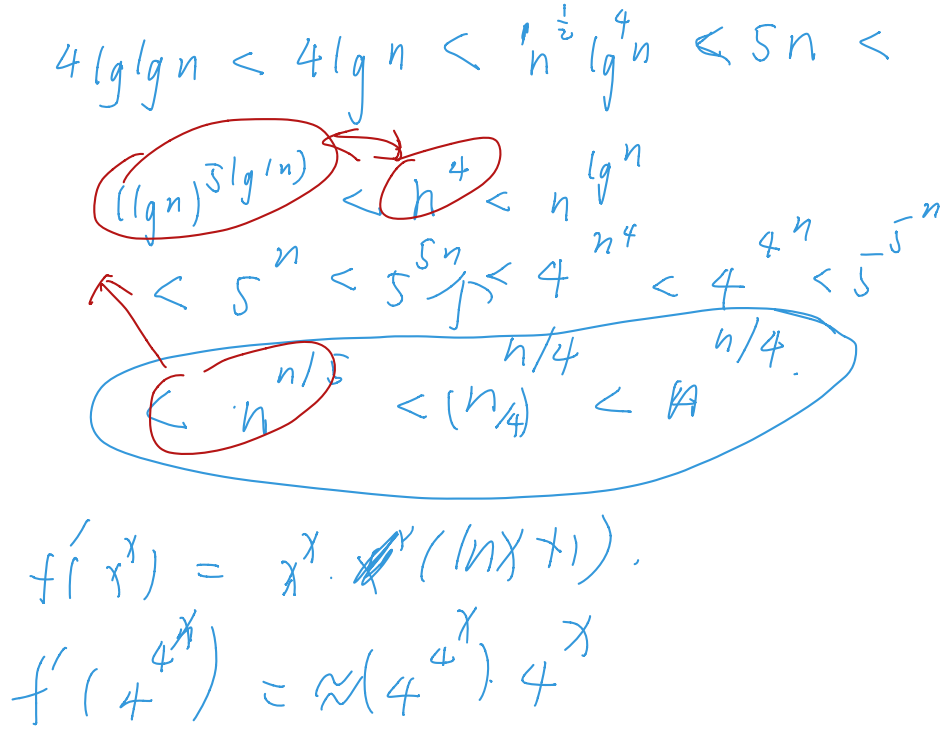
1. A description of the algorithm in English and, if helpful, pseudo-code.
2. A proof (or proof sketch) for the correctness of the algorithm.
3. An analysis of the running time.

We will give full credit only for correct solutions that are described clearly.

Exercise 1-1. Asymptotic Growth

Sort all the functions below in increasing order of asymptotic (big-O) growth. If some have the same asymptotic growth, then be sure to indicate that. As usual, $\lg$ means base 2.

1. $5n$
2. $4 \lg n$
3. $4 \lg \lg n$
4. n^4
5. $n^{1/2} \lg^4 n$
6. $(\lg n)^{5 \lg n}$
7. $n^{\lg n}$
8. 5^n
9. 4^{n^4}
10. 4^{4^n}
11. 5^{5^n}



12. 5^{5n}
13. $n^{n^{1/5}}$
14. $n^{n/4}$
15. $(n/4)^{n/4}$

Exercise 1-2. Solving Recurrences

Give asymptotic upper and lower bounds for $T(n)$ in each of the following recurrences. Assume that $T(n)$ is constant for $n \leq 2$. Make your bounds as tight as possible, and justify your answers.

(a) $T(n) = 4T(n/4) + 5n$
(b) $T(n) = 4T(n/5) + 5n$
(c) $T(n) = 5T(n/4) + 4n$
(d) $T(n) = 25T(n/5) + n^2$
(e) $T(n) = 4T(n/5) + \lg n$
(f) $T(n) = 4T(n/5) + \lg^5 n\sqrt{n}$
(g) $T(n) = 4T(\sqrt{n}) + \lg^5 n$
(h) $T(n) = 4T(\sqrt{n}) + \lg^2 n$
(i) $T(n) = T(\sqrt{n}) + 5$
(j) $T(n) = T(n/2) + 2T(n/5) + T(n/10) + 4n$

Combining the three cases above gives us the following "master theorem".

Theorem 1 *The recurrence*

$$T(n) = aT(n/b) + cn^k$$
$$T(1) = c,$$

where a, b, c, and k are all constants, solves to:

$$T(n) \in \Theta(n^k) \text{ if } a < b^k$$
$$T(n) \in \Theta(n^k \log n) \text{ if } a = b^k$$
$$T(n) \in \Theta(n^{\log_b a}) \text{ if } a > b^k$$

Problem 1-1. Restaurant Location [25 points]

Drunken Donuts, a new wine-and-donuts restaurant chain, wants to build restaurants on many street corners with the goal of maximizing their total profit.

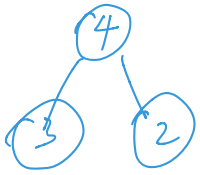
The street network is described as an undirected graph $G = (V, E)$, where the potential restaurant sites are the vertices of the graph. Each vertex u has a nonnegative integer value p_u, which describes the potential ***profit*** of site u. Two restaurants cannot be built on adjacent vertices (to avoid self-competition). You are supposed to design an algorithm that outputs the chosen set $U \subseteq V$ of sites that maximizes the total profit $\sum_{u \in U} p_u$.

First, for parts (a)–(c), suppose that the street network G is acyclic, i.e., a tree.

(a) [5 points]

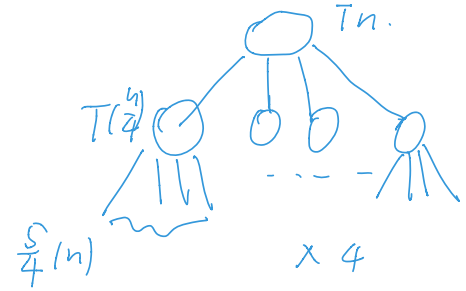
Consider the following "greedy" restaurant-placement algorithm: Choose the highest-profit vertex u_0 in the tree (breaking ties according to some order on vertex names) and put it into U. Remove u_0 from further consideration, along with all of its neighbors in G. Repeat until no further vertices remain.

Give a counterexample to show that this algorithm does not always give a restaurant placement with the maximum profit.



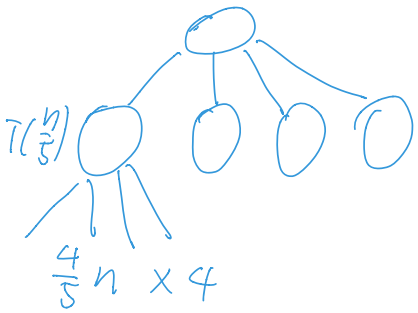
Exercise 1-2.

1)



$\Rightarrow \Theta(n \cdot \lg n)$

(2).



5 n.
$\times (\frac{4}{5})^2$.
$\frac{16}{5}$ n.
$\times (\frac{4}{5})^2$.

Geometric Serieral.

$\Rightarrow \Theta(n)$.

(3). $a=5,\ b=4,\ c=4,\ k=1 \xrightarrow{\text{Case3}} \Theta(n^{\log_4 5})$

(4) $25,\ 5,\ 1,\ 2 \xrightarrow{\text{case2}} \Theta(n^2 \cdot \lg n)$

5) $4,\ 5,\ 1,\ k<1 \xrightarrow{\text{case3}} \Theta(n^{\log_5 4})$

6) $4,\ 5 \Rightarrow \Theta(n^{\log_5 4})$

7) $T(n) = 4 \cdot T(\sqrt{n}) + \lg^5 n$.

$n = 2^m \Rightarrow T(2^m) = 4 \cdot T(2^{m/2}) + m^5$

$S(m) = T(2^m) \Rightarrow S(m) = 4S(\frac{m}{2}) + m^5$

$\Rightarrow T(2^m) = \Theta(m^5) \Rightarrow T(n) = \Theta(\lg^5 n)$

9. $T(n) = T(\sqrt{n}) + 5$

$n = 2^m \Rightarrow T(2^m) = T(2^{m/2}) + 5.$

$S(m) = T(2^m) \Rightarrow S(m) = S(m/2) + 5$

$\Rightarrow S(m) = \lg(m).$

$\Rightarrow T(2^m) = \lg(m).$

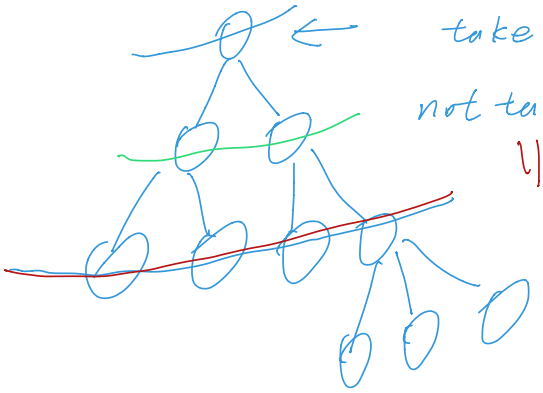
$T(n) = \lg(\lg n)$

10. $T(n) = T(n/2) + T(\frac{n}{5}) + T(\frac{n}{10}) + 4n.$

$\approx 2T(\frac{n}{10}) + 4n.$

$= \Theta(n \cdot \lg n)$

P14. b)



take. P_u + sum(~)

not take. sum(~

1) remember sub prom.

solve by dp

define $A(v)$ is the max profit include node v.

$B(v)$ is the max profit exclude node v.

$$\begin{cases} A(v) = P(v) + \sum_{u \in v.children} (B(u)) \\ B(v) = \sum_{u \in v.children} \max(A(u), B(u)) \end{cases}$$

$A(v) = P(v)$ if v is leaf

$B(v) = 0$

1. dfs sort Nodes in array. N
2. reverse order in N, compute $A(u)$, $B(u)$ recursion
3. Finally. $\max(A(root), B(root))$.

(b) [9 points]

Give an efficient algorithm to determine a placement with maximum profit.

(c) [6 points]

Suppose that, in the absence of good market research, DD decides that all sites are equally good, so the goal is simply to design a restaurant placement with the largest number of locations. Give a simple greedy algorithm for this case, and prove its correctness. like in b, sort by dfs, then include every leaf and delete his parent in tree

(d) [5 points]

Now suppose that the graph is arbitrary, not necessarily acyclic. Give the fastest correct algorithm you can for solving the problem. NP-complete problem. (maximum_independent subset)

Problem 1-2. Radio Frequency Assignment [25 points]

Prof. Wheeler at the Federal Communications Commission (FCC) has a huge pile of requests from radio stations in the Continental U.S. to transmit on radio frequency 88.1 FM. The FCC is happy to grant all the requests, provided that no two of the requesting locations are within Euclidean distance 1 of each other (distance 1 might mean, say, 20 miles). However, if any are within distance 1, Prof. Wheeler will get annoyed and reject the entire set of requests.

Suppose that each request for frequency 88.1 FM consists of some identifying information plus (x, y) coordinates of the station location. Assume that no two requests have the same x coordinate, and likewise no two have the same y coordinate. The input includes two sorted lists, L_x of the requests sorted by x coordinate and L_y of the requests sorted by y coordinate.

(a) [3 points]

Suppose that the map is divided into a square grid, where each square has dimensions $\frac{1}{2} \times \frac{1}{2}$. Why must the FCC reject the set of requests if two requests are in, or on the boundary of, the same square? max distance in a grid is $\frac{\sqrt{2}}{2}$, which < 1.

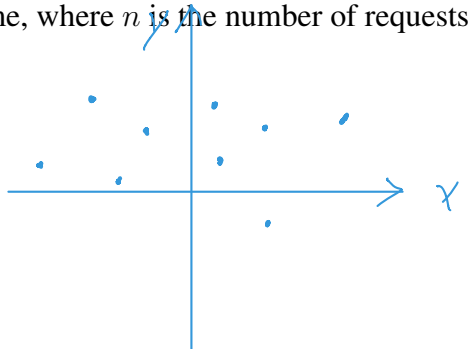
(b) [14 points]

Design an efficient algorithm for the FCC to determine whether the pile of requests contains two that are within Euclidean distance 1 of each other; if so, the algorithm should also return an example pair. For full credit, your algorithm should run in $O(n \lg n)$ time, where n is the number of requests.

Hint: Use divide-and-conquer, and use Part (a).

(c) [8 points]

Describe how to modify your solution for Part (b) to determine whether there are three requests, all within distance 1 of each other. For full credit, your algorithm should run in $O(n \lg n)$ time, where n is the number of requests.



(1) divide by L_x ⇒ left, right.

Fcc(left)
Fcc(right). ⇒ if any not null return.

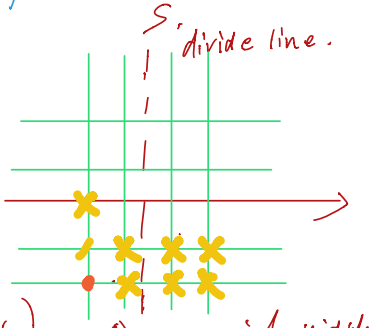
(2) conquer. in size(L) ≤ 2, compute distance.
otherwise divide further.

(3) merge. check request in area S

get request in L. sort by y.

then for r in L.

check. distance(r, next 7 request in L).



Why 7? every request not in grid.

where x is possible for • has distance < 1.

沒懂

C. goon find the pair of request.