COMP7270/7276 MIDTERM I

Instructor: Bo Liu

03/07/2015

Name:
SID:

Instructions:

- You have 75 minutes, so use your time wisely!
- Your only helper is your A-4 size, one-sided, hand-written cheatsheet.
- Have fun!

1 (15 points)

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.

$T(n) = T(n-2) + n^2$
$T(n) = 3T(n/3) + n/lgn$
$T(n) = 3T(n/3 - 2) + n/2$

Q1-(1)

$T(n) = T(n-2) + n^2$

By writing out the sum we see

$$\begin{aligned} T(n) &= T(n-2) + n^2 = T(n-4) + (n-2)^2 + n^2 = \sum_{i=0}^{n/2} (n-2i)^2 \\ &= \sum_{i=0}^{n/2-1} (n-2i)^2 + 0 = \sum_{i=0}^{n/2-1} (n^2 - 4in + 4i^2) = (n/2)n^2 + 4 \sum_{i=0}^{n/2-1} (i^2 - in) \\ &= \frac{n^3}{2} - 4 \sum_{i=1}^{n/2-1} i(n-i) \end{aligned}$$

At this point it's easy to see that $T(n) = O(n^3)$, but we need to do more to make this bound tight. We can split this up into two series

$$T(n) = \frac{n^3}{2} - 4n \sum_{i=1}^{n/2-1} i + 4 \sum_{i=1}^{n/2-1} i^2$$

the first of which is an arithmetic series (see CLRS appendix A) such that

$$\sum_{i=1}^{n/2-1} i = \frac{1}{2}(n/2-1)((n/2-1)+1) = \frac{n}{4}(n/2-1) .$$

So we have that

$$\begin{aligned} T(n) &= \frac{n^3}{2} - 4n\left(\frac{n}{4}(n/2-1)\right) + 4\sum_{i=1}^{n/2-1} i^2 = \frac{n^3}{2} - \frac{n^3}{2} + n^2 + 4\sum_{i=1}^{n/2-1} i^2 \\ &= n^2 + 4\sum_{i=1}^{n/2-1} i^2 \end{aligned}$$

The remaining series (again, see CLRS appendix A) can be written as

$$\sum_{i=1}^{n/2-1} i^2 = \frac{(n/2-1)(n/2-1+1)(2(n/2-1)+1)}{6} = \frac{n}{12} - \frac{n^2}{8} + \frac{n^3}{24}$$

so we see that

$$T(n) = n^2 + 4\left(\frac{n}{12} - \frac{n^2}{8} + \frac{n^3}{24}\right) = \frac{n^3}{6} + \frac{n^2}{2} + \frac{n}{3} = \Theta(n^3) .$$

Even if you didn't know these formulas for series, you could still find the bound by, for example, approximating each by integrals.

Q1-(2)

$T(n) = 3T(n/3) + n/\lg n$.

This is $\Theta(n \lg\lg n)$. By expansion:

$$\begin{aligned}
T(n) &= 3T\left(\frac{n}{3}\right) + \frac{n}{\lg n} \\
&= 3\left(3T\left(\frac{n}{3^2}\right) + \frac{n}{3(\lg n - 1)}\right) + \frac{n}{\lg n} \\
&= 3^2T\left(\frac{n}{3^2}\right) + \frac{n}{\lg n - 1} + \frac{n}{\lg n} \\
&= 3^kT\left(\frac{n}{3^k}\right) + \sum_{i=1}^{k} \frac{n}{\lg n - i + 1} \\
&= n + n\left(1 + \frac{1}{2} + \ldots + \frac{1}{\lg n}\right) \\
&= n + nH_{\log_3 n} = \Theta(n \lg\lg n)
\end{aligned}$$

where H_j denotes the j-th harmonic number and $k = \log_3 n$.

Q 1-(3)

$T(n) = 3T(n/3 - 2) + n/2$

$= \Theta(n \lg n)$. We could guess and substitute:

$$\begin{aligned}
T(n) &\le 3c(n/3-2)\lg(n/3-2) + n/2 \\
&= (cn - 6c)\lg\left(\frac{n-6}{3}\right) + n/2 \\
&= cn\lg\left(\frac{n-6}{3}\right) - 6c\lg\left(\frac{n-6}{3}\right) + n/2 \\
&= cn\lg(n-6) - cn\lg 3 - 6c\lg\left(\frac{n-6}{3}\right) + n/2 \\
&= cn\lg(n-6) + n(\frac{1}{2} - c\lg 3) - 6c\lg\left(\frac{n-6}{3}\right) \\
&\le cn\lg(n-6) - 6c\lg\left(\frac{n-6}{3}\right) \\
&\le cn\lg(n-6) \le cn\lg n
\end{aligned}$$

when $c \ge \frac{1}{2\lg 3}$ and $n \ge 6$. Alternatively, it's easy to see that $T(n) \le 3T(n/3) + n/2$ which is $\Theta(n \lg n)$ by the Master Theorem. Either way, we have that $T(n) = O(n \lg n)$.

In the other direction, we again use induction:

$$\begin{aligned}
T(n) &\ge 3c(n/3-2)\lg(n/3-2) + n/2 \\
&= c(n-6)\lg\left(\frac{n}{3}\left(1 - \frac{6}{n}\right)\right) + \frac{n}{2} \\
&= c(n-6)(\lg n - \lg 3 + \lg(1-6/n)) + n/2 \\
&\ge c(n-6)(\lg n - \lg 3 - 1) + n/2 \qquad \text{for } n \ge 12,\ \lg(1-6/n) \ge -1 \\
&\ge c(n-6)(\lg n - 3) + n/2 \\
&= cn\lg n - 3cn - 6c\lg n + 18c + n/2 \\
&\ge cn\lg n - 3cn - 6cn + 18c + n/2 \\
&= cn\lg n - 9cn + 18c + n/2 \\
&\ge cn\lg n + 18c \qquad \text{for } c \le 1/18 \\
&\ge cn\lg n
\end{aligned}$$

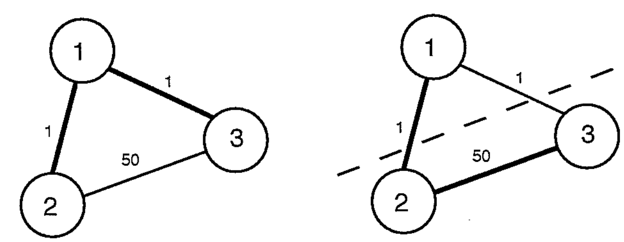
So $T(n) = \Omega(n \lg n)$.

Q-2

▷ **(23.2-8)** *Professor Borden proposes a new divide-and-conquer algorithm for computing minimum spanning trees, which goes as follows. Given a graph $G = (V, E)$, partition the set V of vertices into two sets V_1 and V_2 such that $|V_1|$ and $|V_2|$ differ by at most 1. Let E_1 be the set of edges that are incident only on vertices in V_1, and let E_2 be the set of edges that are incident only on vertices in V_2. Recursively solve a minimum-spanning-tree problem on each of the two subgraphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Finally, select the minimum-weight edge in E that crosses the cut (V_1, V_2), and use this edge to unite the resulting two minimum spanning trees into a single spanning tree.*

Either argue that the algorithm correctly computes a minimum spanning tree of G, or provide an example for which the algorithm fails.

This algorithm does not always find a minimum spanning tree. Consider the following example.



The minimum spanning tree of nodes 1, 2 and 3 has total weight 2, and is shown on the left. Because Professor Borden's algorithm selects a split arbitrarily, it may choose to split the graph such that node 1 is in one set of nodes and nodes 2 and 3 are in another. This leads to a spanning tree of weight 51.

This algorithm does not even necessarily find a spanning tree at all even if one exists—consider the example above but where the edge between nodes 2 and 3 is absent.

Q3

▷ (15.4-5) Longest monotonically increasing subsequence

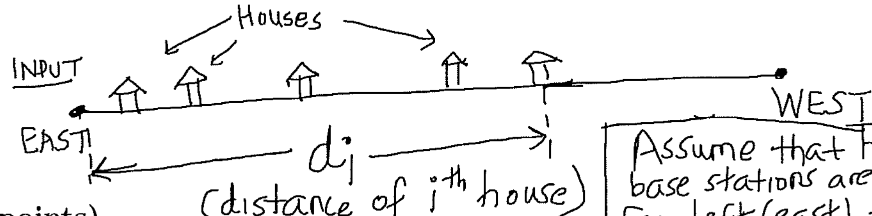
Give an $O(n^2)$-time algorithm to find the longest monotonically increasing subsequence of a sequence of n numbers.

Let X be the input sequence of n numbers. Sort X into non-decreasing order and store as sequence Y. The longest common subsequence between X and Y is the longest monotonically increasing subsequence, which we can find by running LCS(X, Y). That is, on input X:

$Y \leftarrow \textsc{Sort}(X)$
$(b, c) \leftarrow \textsc{Compute-LCS-Table}(X, Y)$
$Z \leftarrow \textsc{Print-LCS}(b, X, |X|, |Y|)$
return Z

Sorting runs in time $O(n \lg n)$ and LCS runs in time $O(n^2)$, so the combined running time is $O(n^2)$.

Q4



Assume that houses & base stations are numbered from left (east) to right (west)

Question ~~1~~ 4: (20 points)

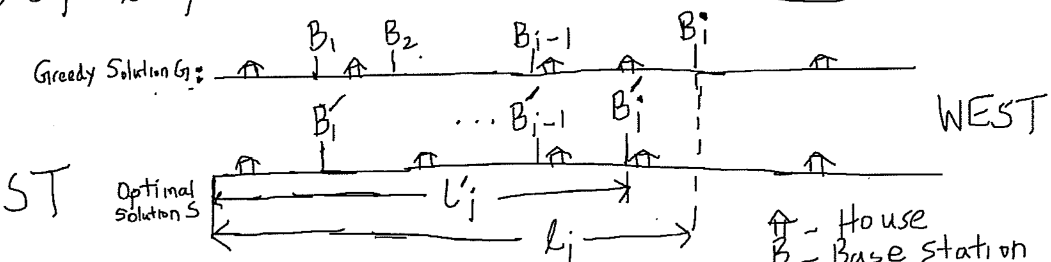
Let's consider a long, straight country road with n houses scattered very sparsely along it. We can picture this road as a long line segment with an eastern endpoint and a western endpoint. Let d_i be the distance of the i^{th} house from the eastern endpoint. We are given as input d_i, for all $1 \leq i \leq n$. Further, let's suppose that despite the bucolic setting, the residents of all these houses are avid cell phone users. You want to place cell phone base stations at certain points along the road, so that every house is within four miles of one of the base stations. Give an efficient algorithm that achieves this goal, using as few base stations as possible.

<u>Algorithm Greedy</u>

1. Input distances $d_1, d_2 \cdots d_{n-1}$, where d_i = distance of the i^{th} house
2. $i \leftarrow 1$
   while $i \leq n$ do
   Place a base station at $l \leftarrow d_i + 4$
   While $d_i \leq l + 4$ do $i \leftarrow i+1$

IDEA: Algorithm Greedy covers houses by placing base stations as far to the right as possible.

<u>Proof of correctness:</u>



For contradiction, assume that greedy Solution G is not optimal. Let G consist of K base stations, $B_1, B_2, \cdots, B_{K-1}, B_K$ ordered from left to right (see figure). Let l_i denote the distance of B_i from the eastern end point, for $1 \leq i \leq K$. If G is not optimal, some other solution with $K' < K$ base stations is optimal. For any two valid solutions, define the index of agreement to be the largest index i such that the two solutions agree on the first i base station locations. Let S be the solution that has the <u>largest</u> index of agreement with G of <u>any</u> optimal solution. We now derive a contradiction by constructing an optimal solution S' with an even larger index of agreement.

<u>CONTINUED ON OVERFLOW PAGE 1</u>

Q4 (continued)

Overflow Page 1

Please use this extra page if you require more space.

Problem 4 continued:

(See Figure on page 2)

Let i denote the index of agreement between G and S. Further, let j denote the index of the first house that is not covered by first i base stations picked by G (or, equivalently by S since both solutions agree on the first i picks). Note that G places the $i+1^{st}$ base station at location $l_{i+1} = d_j + 4$, since G places base stations as far to the right as possible. In particular, the $i+1^{st}$ base station B'_{i+1} of S at location l'_{i+1} must be to the left of the $i+1^{st}$ base station B_{i+1} of G, i.e., $l'_{i+1} < l_{i+1}$. Thus, B_{i+1} covers at least as many houses with indices j or larger as B'_{i+1}. Thus, we can create a new valid optimal solution $S' = S + \{B_{i+1}\} - \{B'_{i+1}\}$ by replacing B'_{i+1} with B_{i+1}. The index of agreement between S' and G is at least $i+1$, i.e., larger than the index of agreement between S and G. Contradiction.

QED

Time Analysis

The while loop of step 2 is iterated at most n times with each iteration taking $O(1)$ time. Thus step 2 takes $O(n)$ time. Step 1 also takes $O(n)$ time. Thus, total time is $O(n)$.