Problem Set for Midterm

Problem 1-3(Substitution, Recursion-tree, Master method)

1. Substitution Method

(1) $T(n)=2T\left(\frac{n}{2}+17\right)+n$ has solution $T(n)=O(n\log_2 n)$

(2) The solution to the recurrence $T(n)=4T\left(\frac{n}{2}\right)+n$ is $T(n)=\Theta(n^2)$.

Show that a substitution proof with the assumption $T(n)\leqslant cn^2$ fails. Then show how to subtract a lower-order term to make a substitution proof work.

(3) The recurrence $T(n)=2T(n-1)+1$ has solution $T(n)=O(2^n)$. Show that a substitution proof with the assumption $T(n)\leqslant c\cdot 2^n$ fails. Then show how to subtract a lower-order term to make a substitution proof work.

2. Recursion-tree Method

(1) Use a recursion tree to determine a good asymptotic upper bound on the recurrence $T(n)=4T\left(\frac{n}{2}+2\right)+n$. Use the substitution method to verify your answer.

(2) Use a recursion tree to determine a good asymptotic upper bound on the recurrence $T(n)=T(n-1)+T\left(\frac{n}{2}\right)+n$. Use the substitution method to verify your answer.

(3) Use a recursion tree to justify a good guess for the solution to the recurrence $T(n)=T(\alpha n)+T((1-\alpha)n)+\Theta(n)$, where α is a constant in the range (0, 1).

3. Master Method

(1) Can the master method be applied to the recurrence $T(n)=4T\left(\frac{n}{2}\right)+n^2\log_2 n$? Why or why not? Give an asymptotic upper bound for this recurrence.

Problem 4(Illustrate the operation of Quicksort on the array A with some numbers.)
(1) See PPT of lecture 6, page 6.

Problem 5-6(Performance analysis of Insertion sort, Merge sort, or Quicksort)
1. Insertion sort
(1) Write the pseudocode of Insertion sort, analyze its best and worst-case time complexity using 'cost and time' concept, illustrate the correctness using loop invariant.
(2) Write pseudocode of Insertion sort with recursive algorithm.

2. Merge sort
(1) Analyze the best and worst-case time complexity of Merge sort and illustrate its correctness using loop invariant.

(2) Although merge sort runs in $\Theta(n \log_2 n)$ worst-case time and insertion sort runs in $\Theta(n^2)$ worst-case time, the constant factors in insertion sort can make it faster in practice for small problem sizes on many machines. Thus, it makes sense to ***coarsen*** the leaves of the recursion by using insertion sort within merge sort when subproblems become sufficiently small. Consider a modification to merge sort in which $\frac{n}{k}$ sublists of length k are sorted using insertion sort and then merged using the standard merging mechanism, where k is a value to be determined.

(a) Show that insertion sort can sort the $\frac{n}{k}$ sublists, each of length k, in $\Theta(nk)$ worst-case time.

(b) Show how to merge the sublists in $\Theta\left(n \log_2\left(\frac{n}{k}\right)\right)$ worst-case time.

(c) Given that the modified algorithm runs in $\Theta\left(nk + n \log_2\left(\frac{n}{k}\right)\right)$ worst-case time, what is the largest value of k as a function of n for which the modified algorithm has the same running time as standard merge sort, in terms of Θ-notation.

3. Quicksort

(1) Analyze the best, worst, and average-case time complexity of Quicksort and illustrate its correctness using loop invariant.

(2) Show that quicksort’ s best-case running time is $\Omega(n\log_2 n)$.

(3) Show that RANDOMIZED-QUICKSORT’s expected running time is $\Omega(n\log_2 n)$.

(4) Suppose that the splits at every level of quicksort are in the proportion $(1-\alpha)$ to α, where $\alpha \in \left(0, \frac{1}{2}\right]$ is a constant. Show that the minimum depth of a leaf in the recursion tree is approximately $-\frac{\log_2 n}{\log_2 \alpha}$ and the maximum depth is approximately $-\frac{\log_2 n}{\log_2 (1-\alpha)}$.(Don’t worry about integer round-off.)

Problem 1-3

1. Substitution Method

(1) $T(n) = 2T(\frac{n}{2}+17) + n$, $T(n) = O(n\log_2 n)$

Solution:

Guess: $T(n) \le cn\log_2 n$ ($c>0$, constant)

Inducive hypothesis: $T(\frac{n}{2}+17) \le c(\frac{n}{2}+17)\log_2(\frac{n}{2}+17)$

Substitution:

$T(n) = 2T(\frac{n}{2}+17) + n$

$\le 2c(\frac{n}{2}+17)\log_2(\frac{n}{2}+17) + n$

$= (cn+34)\log_2(\frac{n}{2}+17) + n$

$\le (cn+34)\log_2 n + n$ ($\frac{n}{2}+17 \le n \Rightarrow n \ge 34$)

However, we cannot prove the guess

by what we got above, we need to change assumption

New guess: $T(n) \le cn\log_2 n - d$ ($d>0$, constant)

New Inducive hypothesis: $T(\frac{n}{2}+17) \le c(\frac{n}{2}+17)\log_2(\frac{n}{2}+17) - d$

Substitution:

$T(n) = 2T(\frac{n}{2}+17) + n$

$\le 2[c(\frac{n}{2}+17)\log_2(\frac{n}{2}+17) - d] + n$

$= (cn+34c)\log_2(\frac{n}{2}+17) - 2d + n$

$= cn\log_2(\frac{n}{2}+17) + 34c\log_2(\frac{n}{2}+17) - 2d + n$

$\le cn\log_2(\frac{3n}{4}) + 34c\log_2(\frac{3n}{4}) - 2d + n$ ($n \ge 68$)

$= cn\log_2 n + cn\log_2(\frac{3}{4}) + 34c\log_2(\frac{3n}{4}) - 2d + n$

$= cn\log_2 n - d + cn\log_2(\frac{3}{4}) + 34c\log_2(\frac{3n}{4}) - d + n$

$\le cn\log_2 n - d$ (if $cn\log_2(\frac{3}{4}) + 34c\log_2(\frac{3n}{4}) - d + n \le 0$)

$\Rightarrow d \ge cn\log_2(\frac{3}{4}) + 34c\log_2(\frac{3n}{4}) + n$

Let $c = \frac{-1}{\log_2(\frac{3}{4})} \Rightarrow cn\log_2(\frac{3}{4}) + 34c\log_2(\frac{3n}{4}) + n = -n - 34 + n = -34$

Thus $\exists\ c = \frac{-1}{\log_2(\frac{3}{4})} > 0$, $d \ge -34$ such that $T(n) \le cn\log_2 n - d$ ($n \ge 68$)

$\Rightarrow T(n) = O(n\log_2 n)$

(2) $T(n) = 4T(\frac{n}{2}) + n$, $T(n) = \Theta(n^2)$

Solution:

Guess: $T(n) \le cn^2$ ($c>0$, constant)

Inducive hypothesis: $T(\frac{n}{2}) \le c(\frac{n}{2})^2$

Substitution:

$T(n) = 4T(\frac{n}{2}) + n$

$\le 4c(\frac{n}{2})^2 + n$

$= cn^2 + n$

Thus, this assumption fails

New guess: $T(n) \le cn^2 - n$

New inducive hypothesis: $T(\frac{n}{2}) \le c(\frac{n}{2})^2 - \frac{n}{2}$

Substitution:

$T(n) = 4T(\frac{n}{2}) + n$

$\le 4[c(\frac{n}{2})^2 - \frac{n}{2}] + n$

$= cn^2 - n$

which shows the subtraction that works.

(3) $T(n) = 2T(n-1) + 1$, $T(n) = O(2^n)$

Solution:

Guess: $T(n) \le c\cdot 2^n$ ($c>0$, constant)

Inducive hypothesis: $T(n-1) \le c\cdot 2^{n-1}$

Substitution:

$T(n) = 2T(n-1) + 1$

$\le 2c\cdot 2^{n-1} + 1$

$= c\cdot 2^n + 1$

Thus, this assumption is fails

New guess: $T(n) \le c\cdot 2^n - n$

New inducive hypothesis: $T(n-1) \le c\cdot 2^{n-1} - (n-1)$

Substitution:

$T(n) = 2T(n-1) + 1$

$\le 2[c\cdot 2^{n-1} - (n-1)] + 1$

$= c\cdot 2^n + 1 - 2(n-1)$

$= c\cdot 2^n + 3 - 2n$

$= c\cdot 2^n - n + 3 - n$

$\le c\cdot 2^n - n$ ($3 - n \le 0 \Rightarrow n \ge 3$)

Thus, this subtraction works.

---

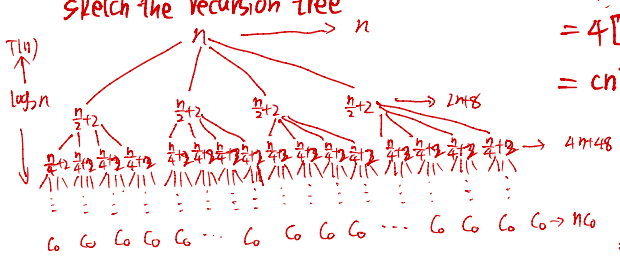
2. Recursion tree Method

(1) $T(n) = 4T(\frac{n}{2}+2) + n$

Solution:

Rewrite $T(n) = \begin{cases} c_0 & n=1 \\ 4T(\frac{n}{2}+2)+n & n>1 \end{cases}$

Sketch the recursion tree



Thus, the total cost is

$T(n) = \sum_{i=0}^{\log_2 n - 1} 4^i \frac{n}{2^i} + \sum_{i=1}^{\log_2 n - 1} 4^i \cdot 2 + c_0 n^2$

$= n(n-1) + \frac{1}{3}(\frac{n^2}{4} - 1) + c_0 n^2$

$= (c_0 + \frac{7}{6})n^2 - n - \frac{2}{3}$

$= O(n^2)$

Use substitution method to verify

Guess: $T(n) \le cn^2$ ($c>0$, constant)

Inducive hypothesis: $T(\frac{n}{2}+2) \le c(\frac{n}{2}+2)^2$

Substitution:

$T(n) = 4T(\frac{n}{2}+2) + n$

$\le 4c(\frac{n}{2}+2)^2 + n$

$= cn^2 + (8c+1)n + 16c$

Thus, this assumption fails

New guess: $T(n) \le cn^2 - dn$ (d is constant)

New inducive hypothesis: $T(\frac{n}{2}+2) \le c(\frac{n}{2}+2)^2 - d(\frac{n}{2}+2)$

Substitution:

$T(n) = 4T(\frac{n}{2}+2) + n$

$\le 4[c(\frac{n}{2}+2)^2 - d(\frac{n}{2}+2)] + n$

$= 4[\frac{c}{4}n^2 + (2c - \frac{d}{2})n + 4c - 2d] + n$

$= cn^2 + (8c - 2d + 1)n + 16c - 8d$

$= cn^2 - dn + (8c - d + 1)n + 16c - 8d$

$\le cn^2 - dn$ (if $8c - d + 1 \le 0$ $\Rightarrow d \ge 8c + 1$)

$\Rightarrow T(n) = O(n^2)$

(2) $T(n) = T(n-1) + T(\frac{n}{2}) + n$

Solution: From the recurrence we know this tree is asymetric, with two heights $h_1 = n-1$, $h_2 = \log_2 n$, where $h_2 < h_1$, indicating that h_1 is the longest path of this tree.

Sketch the recursion tree

To find the lower bound, summing the costs on the longest path

$\sum_{i=0}^{n-1}(n-i) = \frac{1}{2}n(n+1) = \frac{1}{2}n^2 + \frac{1}{2}n$

$\Rightarrow T(n) \ge \frac{1}{2}n^2 + \frac{1}{2}n \Rightarrow T(n) \ge \frac{1}{2}n^2$

Thus, $T(n) = \Omega(n^2)$

To find upper bound

Since $T(n) = T(n-1) + T(\frac{n}{2}) + n \le 2T(n-1) + n$

Consider $T'(n) = 2T(n-1) + n$

$\Rightarrow T'(n) = \sum_{i=0}^{n-1} 2^i(n-i) = 2^{n+1} - n - 2$

$= O(2^n)$

Use substitution method to verify (in terms of $T'(n)$)

Guess: $T'(n) \le c\cdot 2^n$ ($c>0$, constant)

Inducive hypothesis: $T'(n-1) \le c\cdot 2^{n-1}$

Substitution

$T'(n) = 2T(n-1) + n$

$\le 2c\cdot 2^{n-1} + n$

$= c\cdot 2^n + n$

This assumption fails,

New guess: $T'(n) = c\cdot 2^n - dn$ ($d>0$, constant)

New inducive hypothesis: $T'(n-1) = c\cdot 2^{n-1} - d(n-1)$

Substitution

$T'(n) = 2T(n-1) + n$

$\le 2[c\cdot 2^{n-1} - d(n-1)] + n$

$= c\cdot 2^n - 2d(n-1) + n$

$= c\cdot 2^n - dn + n + 2d - dn$

$\le c\cdot 2^n - dn$

$n + 2d - dn \le 0$

$d(n-2) \ge n$

$d \ge \frac{n}{n-2} = \frac{n\cdot 2 + 2}{n-2} = 1 + \frac{2}{n-2}$

(3) $T(n)=T(\alpha n)+T((1-\alpha)n)+\Theta(n)\ (0<\alpha<1)$

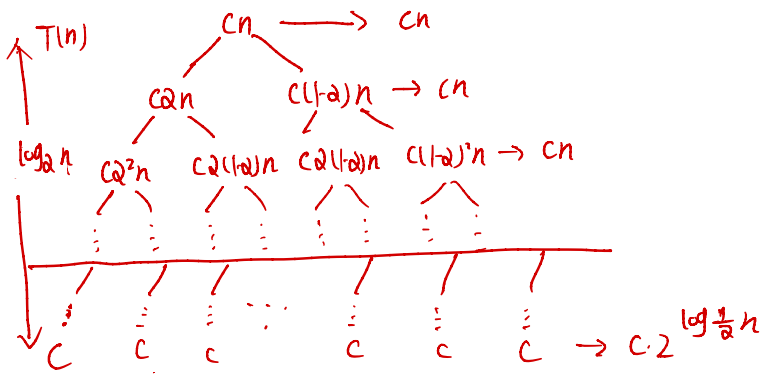
Solution:

Rewrite $T(n)=\begin{cases} c & n=1 \\ T(\alpha n)+T((1-\alpha)n)+cn & n>1 \end{cases}$

Solve the depth of two paths: $h_1=\log_{\frac{1}{\alpha}} n$, $h_2=\log_{\frac{1}{1-\alpha}} n$

Case 1: $h_1 \geq h_2 \Rightarrow \frac{1}{2} \leq \alpha < 1$, which indicates h_1 is the longest path of the tree

Sketch recursion tree



Let the sum of cost on the tree between depth (h_2+1) and (h_1-1) be m

$$\Rightarrow T(n)=\sum_{i=0}^{\log_{\frac{1}{1-\alpha}} n} cn + m + c\cdot 2^{\log_{\frac{1}{\alpha}} n}$$

$$= cn\log_{\frac{1}{1-\alpha}} n + m + cn^{\log_{\frac{1}{\alpha}} 2}$$

$$\leq cn\log_{\frac{1}{1-\alpha}} n + cn + cn^{\log_{\frac{1}{\alpha}} 2}$$

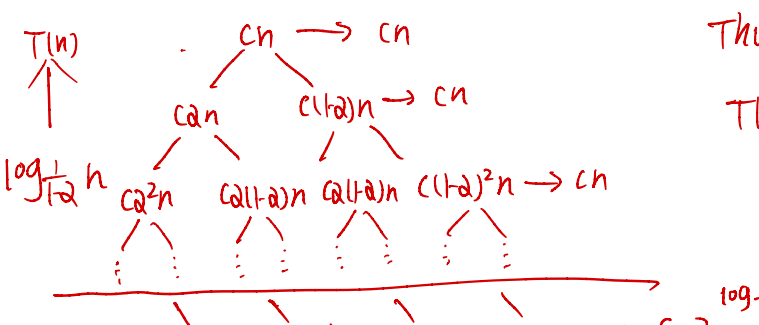
$$\leq cn\log_2 n + cn + cn^{\log_{\frac{1}{\alpha}} 2}$$

$$= O(n\log_2 n)$$

Thus, $T(n)\leq O(n\log_2 n)$

To find lower bound, assume $h_2 \geq h_1\ (0<\alpha\leq\frac{1}{2})$:

Sketch recursion tree



Let the sum of cost on the tree between depth (h_1+1) and h_2 be m_1

$$\Rightarrow T(n)=\sum_{i=0}^{\log_{\frac{1}{\alpha}} n} cn + m_1$$

$$= cn\log_{\frac{1}{\alpha}} n + m_1$$

$$\geq cn\log_{\frac{1}{\alpha}} n + (\log_{\frac{1}{1-\alpha}} n - \log_{\frac{1}{\alpha}} n)cn$$

$$= cn\log_{\frac{1}{1-\alpha}} n$$

$$\geq cn\log_2 n$$

$$= \Omega(n\log_2 n)$$

Thus, $T(n)=\Omega(n\log_2 n)$

Therefore, $T(n)=\Theta(n\log_2 n)$

3. Master method

(1) $T(n)=4T(\frac{n}{2})+n^2\log_2 n$

Solution:

we have $n^{\log_b a}=n^{\log_2 4}=n^2$

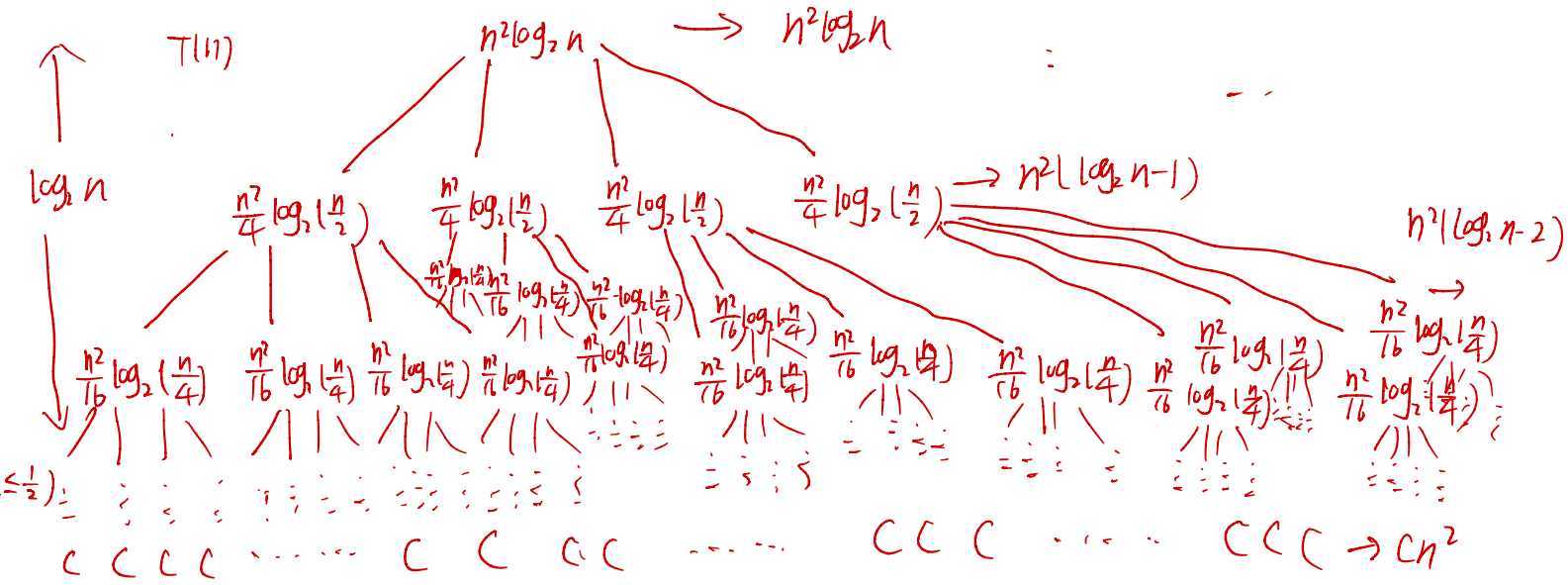
$f(n)=n^2\log_2 n$

Thus, there is no way expressing $f(n)$ as $O(n^{2-\varepsilon})$ and $\Omega(n^{2+\varepsilon})$ $(\varepsilon>0)$

We cannot use master method to solve this recurrence.

Rewrite $T(n)=\begin{cases} c & n=1 \\ 4T(\frac{n}{2})+n^2\log_2 n & n>1 \end{cases}$

Sketch recursion tree



Thus, the total cost of the tree is

$$T(n)=\sum_{i=0}^{\log_2 n-1} 4^i\cdot\frac{n^2}{4^i}\log_2\left(\frac{n}{2^i}\right)+cn^2$$

$$=\sum_{i=0}^{\log_2 n-1} n^2\log_2\left(\frac{n}{2^i}\right)+cn^2$$

$$=n^2\sum_{i=0}^{\log_2 n-1}(\log_2 n-i)+cn^2$$

$$=\frac{n^2}{2}\left[(\log_2 n)^2+\log_2 n\right]+cn^2$$

$$=\frac{1}{2}n^2(\log_2 n)^2+cn^2+\frac{1}{2}n\log_2 n$$

$$=O(n^2\log_2^2 n)$$

$$\Rightarrow T(n)=O(n^2\log_2^2 n)$$