Design and Analysis of Algorithms
Summer 2017
Assignment 6
June 29th, 2017

---

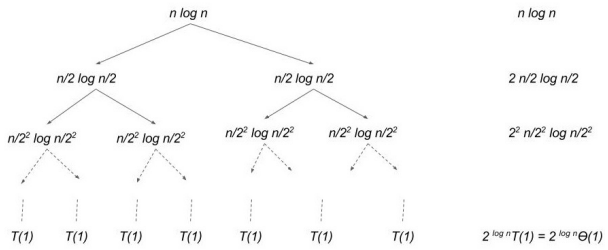
1. (a) (20 points) For each of the following recurrences, first check if master theorem is applicable, and if yes, calculate the complexity using it.

   1. $T(n) = 2t(\frac{n}{2}) + n \log n$
   2. $T(n) = 0.3T(\frac{n}{2}) + \frac{1}{n}$
   3. $T(n) = 2T(\frac{n}{2}) + n^3$
   4. $T(n) = 7^n T(\frac{n}{2}) + n^4$
   5. $T(n) = 4T(\frac{n}{2}) + n^2\sqrt{n}$
   6. $T(n) = 64T(\frac{n}{8}) - n^2 \log n$
   7. $T(n) = 16T(\frac{n}{4}) + n^2$

   Solution:

   1. It's a recurrence with $a = 2, b = 2$. Since $f(n) = n \lg n = \Theta(n^{\lg_b a} \lg^k n) = \Theta(n^{\lg_2 2} \lg n)$ with $k = 1$, case 2 applies and $T(n) = \Theta(n \lg^2 n)$.
   2. $a \ngeq 1$, we cannot apply master theorem
   3. $T(n) = 2T(n/2) + n^3 = \Theta(n^3)$. This is a divide-and-conquer recurrence with $a = 2$, $b = 2$, $f(n) = n^3$, and $n^{\log_b a} = n^{log_2 2} = n$. Since $n^3 = \Omega(n \log_2 2 + 2)$ and $a/b^k = 2/2^3 = 1/4 < 1$, case 3 of the master theorem applies, and $T(n) = \Theta(n^3)$.
   4. a is not constant, master theorem doesn't apply
   5. We have $f(n) = n^2\sqrt{n} = n^{5/2}$ and $n^{\lg_b a} = n^{\log_2 4} = n^{\lg 2}$. Since $n^{5/2} = \Omega(n^{\lg 2+3/2})$, we look at the regularity condition in case 3 of the master theorem. We have $af(n/b) = 4(n/2)^2\sqrt{n/2} = n^{5/2}/\sqrt{2} \leq cn^{5/2}$ for $1/\sqrt{2} \leq c < 1$. Case 3 applies, and we have $T(n) = \Theta(n^2\sqrt{n})$
   6. Not in the $aT(n/b) + f(n)$ form, Master theorem doesn't apply
   7. $a = 16, b = 4, f(n) = n^2$, and $n^{log_b a} = n^{log_4 16} = n^2$. Since $n^2 = \Theta(n^{\log_4 16})$, case 2 of the master theorem applies, and $T(n) = \Theta(n^2 \lg n)$.

   (b) (10 points) Confirm the result for recurrence 1 using recursion tree



Solution:

$$\begin{aligned}
T(n) &= \sum_{i=0}^{\lg n-1} 2^i \frac{n}{2^i} \lg \frac{n}{2^i} + 2^{\lg n} \\
&= \sum_{i=0}^{\lg n-1} n \lg \frac{n}{2^i} + n \\
&= \sum_{i=0}^{\lg n-1} n(\lg n - i) + n \\
&= n \sum_{i=1}^{\lg n} i + n \\
&= n\Theta(\lg^2 n) + n \\
&= \Theta(n \lg^2 n)
\end{aligned}$$

(c) (10 points) Confirm the result for recurrence 3 using induction.

Solution: We need to prove that there is some n_0 and c such that for any $n \geq n_0$, we have $T(n) \leq cn^3$

Base case: $T(1) \leq c.1^3 = c$, this can be satisfied if c is chosen to be greater than $T(1)$

Assumption: $T(m) \leq cm^3$ for all $m < n$

Induction step: $T(n) \leq cn^3$

$T(n) = 2T(n/2) + n^3 \leq 2c(n/2)^3 + n^3 \leq cn^3$. This can be satisfied if $(c/4)n^3 + n^3 \leq cn^3$, or, $c/4 + 1 \leq 1$, which means $c \geq 4/3$. So we can summarize that $T(n) \leq cn^3$ for a choice of c that $c \geq \max(T(1), 4/3)$

2. (10 points) Given an array of numbers, find a peak in the array, i.e, an element which is greater than or equals to the its neighbors (previous and the next number). The first and last elements have only one neighbor.

   Example: Input: 2 3 5 4 6 7 3

   Returns: Either 5 or 7 (one is enough)

   Solution: A summarized solution is explained in the following page `http://www.geeksforgeeks.org/find-a-peak-in-a-given-array/`

   For more detailed explanation take a look at the *references* section in the website, there is a video and a good reference that covers both this question and also the next one:

   `http://courses.csail.mit.edu/6.006/spring11/lectures/lec02.pdf`

3. (15 points) . Now Extend the previous question to a 2D array: find an element that is greater than or equals to its neighbors (diagonal adjacent are not counted as neighbors, so each element has at most 4 neighbors to consider)

   Solution: `http://www.geeksforgeeks.org/find-peak-element-2d-array/` Similar to the previous question, there is a link in the references that have more explanation for this part of the question, as well as the previous part: `https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-006-introduction-to-algorithms-fall-2011/lecture-videos/MIT6_006F11_lec01.pdf`

4. (a) (5 points) Explain the traditional polynomial multiplication, and state its complexity

   (b) (15 points) Improve the complexity using Divide and Conquer method.

   Solution: This is almost the same problem as integer multiplication, you replace 2 with x and you would get an algorithm for polynomial multiplication. However, you can look into this pdf `http://algorithm.cs.nthu.edu.tw/~course/Extra_Info/Divide%20and%20Conquer_supplement.pdf`, which derives the same thing, but focusing on polynomial multiplication instdead of integer multiplications.

5. Given an array $A[1, ..., n]$ of integers, design an algorithm that returns a pair $i < j$ so that that $A[j] - A[i]$ is maximum.

   (a) (10 points) Design an $O(n \lg n)$ algorithm

   (b) (10 points) Design and $O(n)$ algorithm

Solution: This is another famous question, which I formed an abstract question from. It is usually worded in stock market terminology and has been discussed in many resources, such as Kleingberg's book. I found the following solution very well written: `https://stackoverflow.com/questions/7086464/maximum-single-sell-profit`

6. You are given a sorted list of 2k+1 shoes with their product identification numbers (pid). The list includes k pairs and one single shoe. Each pair of shoes have the same pid. Design an algorithm that finds the single id in the list:

   (a) (5 points) Design an $O(n)$ algorithm

   (b) (10 points) Design and $O(\lg n)$ algorithm

   Example Input: 5 5 12 12 18 18 19 21 21 50 50

   Output: 19

Solution: Another well know problem, that I just *tried* to create a story for. A [very] detailed solution can be found here: `https://web.stanford.edu/class/archive/cs/cs161/cs161.1138/handouts/080%20Guide%20to%20Divide-and-Conquer.pdf`