Basic Algorithms CSCI-UA.0310

Homework 4

Due: March 1st, 4 PM EST

Instructions

Please answer each **Problem** on a separate page. Submissions must be uploaded to your account on Gradescope by the due date and time above.
Please note that no late submission will be accepted for this homework.

Problems to submit

Problem 1 (13+12 points)

Recall that in the deterministic approach for choosing the pivot in the SELECT algorithm, first we split the elements of the array into groups of size 5. We want to see how the running time of SELECT changes by varying the size of groups. Consider the following cases:

(a) Split the array into groups of size 3

(b) Split the array into groups of size 7

For each case, write the corresponding recursion for the running time of the SELECT algorithm, draw the corresponding recursion tree for the worst case, and use it to solve the recursion for the worst case.
For full credit, your answers must use $\Theta(.)$ notation (You do NOT need to use strong induction to prove your result).

Problem 2 (25 points)

Considering again the the deterministic approach for choosing the pivot in the SELECT algorithm, we recursively called the SELECT algorithm on the $n/5$ chosen medians, i.e. each of the chosen $n/5$ elements is a median of 5 elements of the input array.
We want to see how the running time of SELECT changes if we instead choose $n/5$ arbitrary elements of the input array, i.e., the $n/5$ chosen elements are not chosen as the medians. Consider the following modified version of the SELECT algorithm:

- Choose the first $n/5$ elements of the input array A, i.e., $A[1 \dots n/5]$
- Find their median by recursively calling SELECT, i.e., SELECT$(A[1 \dots n/5],\ n/10)$
- Set the pivot as their median
- Use this pivot with the rest of the SELECT algorithm covered in the lecture (with little modification)

(a) Show that there are at least $n/10$ elements of A less than the pivot and there are at least $n/10$ elements of A greater than the pivot.

(b) Write the recursion for the running time of this modified SELECT algorithm and draw the corresponding recursion tree for the worst case.

(c) Express the running time of the recursion tree (worst case) as a sum.

Problem 3 (25 points)

We learnt how to solve recursions by using a recursion tree based approach. In this exercise, we learn another approach to solve recursions: using the *Master Theorem*.

Theorem 1. *(Master Theorem) Consider the following recursion*

$$T(n) = aT(n/b) + f(n),$$

with the constants $a \geq 1$ and $b > 1$ and the positive function $f(n)$ (i.e., $f(n) > 0$ for all n).
There are three cases:

(a) If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.

(b) If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log n)$.

(c) If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large values of n, then $T(n) = \Theta(f(n))$.

Some remarks regarding the Master Theorem:

- The Master Theorem can be proved by drawing the corresponding recursion tree but we do not cover it for the sake of simplicity.
- Note that not all the recursions can be solved by using the Master Theorem (see Example III below).

Let's see some examples:

Example I: Consider the following recursion

$$T(n) = 4T(n/2) + n^{1.5}.$$

We have $a = 4$ and $b = 2$. Choosing $\epsilon = 0.1$ (or any value of ϵ satisfying $\epsilon \leq 0.5$), we get

$$f(n) = n^{1.5} = O(n^{\log_b a - \epsilon}).$$

Thus, according to Case (a) of the Master Theorem, we have $T(n) = \Theta(n^{\log_b a}) = \Theta(n^2)$.

Example II: Consider the following recursion

$$T(n) = 2T(n/2) + n^{1.5}.$$

We have $a = 2$ and $b = 2$. Choosing $\epsilon = 0.1$ (or any value of ϵ satisfying $\epsilon \leq 0.5$), we get

$$f(n) = n^{1.5} = \Omega(n^{\log_b a + \epsilon}).$$

Also, choosing $c = 1/\sqrt{2}$ (note that $c < 1$), we get

$$2f(n/2) \leq cf(n),$$

for all n (note that $\leq$ can be even replaced by $=$).
Thus, according to Case (c) of the Master Theorem, we have $T(n) = \Theta(f(n)) = \Theta(n^{1.5})$.

Example III: The Master Theorem cannot be applied to the following recursion (why?)

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

For each of the following recursions, if it can be solved with the Master Theorem, use the Theorem to find the explicit answer for $T(n)$ (In this case, for full credit, your answers must use $\Theta(.)$ notation and you should check that all conditions of the Master Theorem apply as discussed in the Examples above). Otherwise, fully justify why the Master Theorem does not apply.

(a) $T(n) = 4T(n/2) + n^2$

(b) $T(n) = 2T(n/2) + \log n$

(c) $T(n) = 0.2T(n/2) + n$

(d) $T(n) = 8T(n/2) + 2^n$

(e) $T(n) = 2T(n/2) + \frac{n}{\log n}$

Problem 4 (12+13 points)

(a) Given the input array A of size n with distinct elements and the positive integer k satisfying $1 \leq k \leq n$, modify the SELECT algorithm to develop an $O(n)$ time algorithm to find all the kth smallest elements of A. In other words, your algorithm must output the 1st smallest, and the 2nd smallest, ..., and the kth smallest elements of A.
You should write the pseudo-code of your algorithm.

(b) Given the input array A of size n with distinct elements and the positive integers k, l, satisfying $1 \leq l \leq k \leq n$, develop an $O(n)$ time algorithm to find the lth smallest, and the $(l+1)$th smallest, ..., and the kth smallest elements of A (Thus, your algorithm must return $k - l + 1$ elements).
You do NOT need to write the pseudo-code of your algorithm.

Bonus problem

Bonus Problem 1

Consider the following recursion

$$T(n) = 3T\left(\frac{n-\sqrt{n}}{2}\right) + n^2.$$

(a) Show that the Master Theorem cannot be directly applied to solve $T(n)$.

(b) Replace $\frac{n-\sqrt{n}}{2}$ by a simplified upper bound. Choose the upper bound such that the resulting recursion can be solved by the Master Theorem. The result of this modified recursion gives an upper bound for $T(n)$.

(c) Repeat part (b) to find a lower bound for $T(n)$:
Replace $\frac{n-\sqrt{n}}{2}$ by a simplified lower bound for sufficiently large values of n. Choose the lower bound such that the resulting recursion can be solved by the Master Theorem. The result of this modified recursion gives a lower bound for $T(n)$.

(d) Can you choose the upper and lower bounds for $\frac{n-\sqrt{n}}{2}$ in such a way that the resulting upper and lower bounds for $T(n)$ obtained in parts (b) and (c) match? If so, this gives an answer to $T(n)$ which is of the form $\Theta(.)$.