SIENAcollege
Computer Science

Computer Science 385
Design and Analysis of Algorithms
Siena College
Spring 2018

Problem Set 4
Due: 4:00 PM, Wednesday, March 14, 2018

You may work alone or in a group of 2 or 3 on this assignment. However, in order to make sure you learn the material and are well-prepared for the exams, you should work through the problems on your own before discussing them with your partner, should you choose to work with someone. In particular, the "you do these and I'll do these" approach is sure to leave you unprepared for the exams.

Groups must be formed and "registered" as a group by an email to *jteresco@siena.edu* by 4:00 PM, Wednesday, March 7, 2018. This applies to those who choose to work alone as well!

There is a significant amount of work to be done here, and you are sure to have questions. It will be difficult if not impossible to complete the assignment if you wait until the last minute. A slow and steady approach will be much more effective.

Submitting

Please submit a hard copy (typeset preferred, handwritten OK but must be legible) for all written questions. Only one submission per group is needed.

Homework Set Problems

Question 1:
Determine closed forms for each of the following recurrence formulas using backward substitution. (24 points)

For each problem: (i) show at least the first 3 substitutions (for 4 points), (ii) show the pattern for the i^{th} substitution (2 points), (iii) state the value that i will have in the substitution that allows you to apply the base case (1 point), (iv) give the closed form (4 points), and (v) give the Θ efficiency class of the recurrence (1 point).

(a) $T(n) = T(n-1) + 5$ for $n > 1, T(1) = 3$

(b) $T(n) = T(n-1) + 5n$ for $n > 1, T(1) = 3$

(c) $T(n) = T(n/2) + 4$ for $n > 1, T(1) = 1$

(d) $T(n) = T(n/2) + 4n$ for $n > 1, T(1) = 1$

(e) $T(n) = 2T(n/2) + 3$ for $n > 1, T(1) = 1$

(f) $T(n) = 2T(n/2) + 3n$ for $n > 1, T(1) = 1$

(g) $T(n) = 2T(n/2) + an$ for $n > 1$ and any positive constant a, $T(1) = 1$

(h) $T(n) = 2T(n/2) + n^2$ for $n > 1$, $T(1) = 1$

Note: although your final closed form for (g) above will include the constant a, your Θ growth rate should only be in terms of n.

I will choose 2 of the 8 problems above to grade out of 12 points each. Any omitted problem will be chosen automatically (remember, I hate grading), so do them all!

For the next four questions, consider the problem of counting, in a given array of integers, the number of subarrays that start with a `0` and end with a `1`. For example, there are four such subarrays in `[2, 0, 1, 0, 0, 8, 1, 9, 0]`.

Question 2:
Write pseudocode for a brute-force algorithm to solve this problem, starting with the pseudocode below. (8 points)

ALGORITHM COUNT01SUBARRAYS(A)
//Input: an array of integers $A[0..n-1]$
//Output: the number of subarrays starting with 0 and ending with 1

Question 3:
Give the efficiency class of your algorithm, and briefly justify your answer. (2 points)

Question 4:
Improve upon this by writing a **linear time** algorithm solving this problem, starting with the same psuedocode as above. (8 points)

Question 5:
Briefly justify why your algorithm runs in linear time. (2 points)

For the remaining questions, consider the following problem.

Determining the best time to buy and sell a stock is a problem of interest to many people for obvious reasons. Perhaps examining past behavior might be a way to predict future behavior, leading to massive profits. The problem we will consider is to find optimal buy and sell days given a sequence of known daily stock price changes over some period of days in the past. For example, consider the daily changes in value shown below for Roger Bacon Enterprises stock over 15 days. The values indicate the amount by which the stock price changed from the start of the trading day to the end of the trading day. Positive values indicate the stock increased in price on that day, and negative values indicate the price fell.

```
-6 -1 +5 +7 +5 +3 -5 -7 -1 +10 +1 +5 -20 +10 +1
```

We want to calculate the maximum amount of profit you could have earned assuming you buy exactly once (at the beginning of a day) and sell exactly once (at the end of a day, possibly the same day you bought) during the 15 day period. For example, if you bought on day 2 and sold on day 5, then you would have a net profit of 20. However, if you bought on day 2 and sold on day 11, then you would have a net profit of 23. (Note: we're all computer scientists here, so the first day is numbered 0.)

We could easily solve this problem in $\Theta(n^2)$ time using brute force by examining all pairs of buy and sell days, where n is the number days in the period being considered. But that's last month's news. We can do better!

Question 6:
Improve upon this by giving a divide and conquer algorithm solving this problem that runs asymptotically faster than $\Theta(n^2)$. Start with the pseudocode below. It returns the maximum profit you could have made if you bought and sold exactly once within the range of days indicated. A complete and correct base case is provided. (10 points)

ALGORITHM MAXPROFIT($A, sday, eday$)
//Input: an array of price changes $A[0..n-1]$
//Input: start ($sday$) and end ($eday$) days of the subrange to consider
//Output: the maximum possible profit from buying and selling within the range
if $sday = eday$ **then**
// only one day in the range, so must buy and sell on this day
return $A[sday]$

Question 7:
Once you have your algorithm, complete the recursive call tree started below, showing all the recursive calls your algorithm will make and the return value from each. (4 points)

The root of your tree should look like what you see below, with `MP` (short for `MaxProfit`) and the blank filled in with the answer that the call returns. All other nodes in your tree should be formatted similarly.

`MP(A,0,7)`
`returns` _____

Use `A = [6, -50, 2, 13, 5, -14, 1, 6]` as your input array.

Question 8:
Give a recurrence formula for the worst case running time of your algorithm, and then give its Θ efficiency class. (You may find the closed form of your recurrence in any way you like and you do not need to show your work in getting it.) (2 points)