Selected Problems in CLRS

Section 4

Notifications

Problem Difficulty (count with star)

1. you can solve w/o the brain
2. you can solve if you think a bit
3. you can solve if you think carefully
4. you might solve if you push yourself
5. you can solve if you use other's brain

Exercise

4.1-1 $\star\star$

What does FIND-MAXIMUM-SUBARRAY return when all elements of A are negative?

4.1-3 $\star\star$

Implement both the brute-force and recursive algorithms for the maximum-subarry problem on your own computer.

> **Note :** There are template for this problem in `Algo/src/algo.cpp`

4.1-4 $\star\star$

How would you change any of the algorithms that do not allow empty subarrays to permit an empty subarray to be the result?

4.1-5 $\star\star\star\star$ ($\star$ if you know it as ad-hoc)

Write non-recursive, linear-time algorithm for the maximum-subarray problem.

> **Note :** Skip ***4.2*** which deal with strassen

4.3-1 $\star\star$ (**solve fomally**)

Show that the solution of $T(n) = T(n-1) + n$ is $O(n^2)$

4.3-3 $\star\star\star$ (**solve fomally**)

Show that $T(n) = 2T(\lfloor n/2 \rfloor) + n$ is $\Theta(n \lg n)$

4.3-6 $\star\star\star$ (**solve fomally**)

Show that the solution to $T(n) = 2T(\lfloor n/2 \rfloor + 17) + n$ is $O(n \lg n)$

4.3-7 $\star\star\star$ (**solve fomally**)

Show that the solution to $T(n) = 4T(n/3) + n$ is $T(n) = \Theta(n^{\log_3 4})$ without the master method

4.3-9 $\star\star\star$ (**solve fomally**)

Solve the recurrence $T(n) = 3T(\sqrt{n}) + \log n$. Your solution should be asymtotically tight.

4.4-3 $\star\star$

Use a recursion tree to determine a good asymptotic upper bound on the recurrence $T(n) = T(n/2) + n^2$. Use the substitution method to verify your answer.

4.4-5 $\star\star\star$

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

4.4-8,9 $\star\star$

Use a recursion tree to give an asymptotically tight solution to the recurrence

$$T_1(n) = T_1(n-a) + T_1(a) + cn \quad T_2(n) = T_2(\alpha n) + T_2((1-\alpha)n) + cn$$

where $a \geq 1$, $c > 0$ and $0 < \alpha < 1$ are constants.

4.5-5 $\star\star\star$

Consider the regularity condition $af(n/b) \leq cf(n)$ for some constant $c < 1$, which is part of case 3 of master theorem. Give an example of constnat $a \geq 1$ and $b > 1$ and a function $f(n)$ that satisties all the conditions in case 3 of the master theorem except the regularity condition.

> **Note :** think carefully why case 3 needs regularity condition.

4.6-3 $\star\star\star\star$

Show that case 3 of the master theorem is overstated, in the sense that the regularity condition $af(n/b) \leq cf(n)$ for some constant $c < 1$ implies that there exists a constant $\epsilon > 0$ such that $f(n) = \Omega(n^{\log_b a+\epsilon})$

Problems

4-3 $\star\star\star$

1. $T(n) = 4T(n/3) + n\lg n$
2. $T(n) = 3T(n/3) + n/\lg n$
3. $T(n) = 4T(n/2) + n^2\sqrt{n}$
4. $T(n) = 3T(n/3 - 2) + n/2$
5. $T(n) = 2T(n/2) + n/\lg n$
6. $T(n) = T(n/2) + T(n/4) + T(n/8) + n$
7. $T(n) = T(n-1) + 1/n$
8. $T(n) = T(n-1) + \lg n$
9. $T(n) = T(n-2) + 1/\lg n$
10. $T(n) = \sqrt{n}T(\sqrt{n}) + n$