Algorithms & Data Structures 2024/25

Practical Week 10

I realise that model answers tend to be easily available. Please however try to come up with your own solutions. If you do get stuck then ask the demonstrators for help.

1. Finish anything that may be left over from last week.
2. Simulate by hand the variant of QuickSort/Partition as covered in class on a "random" input of size 32.

Solution:

Omitted.

3. Solve the following recurrences using the Master Theorem, or state why it cannot be applied.

(a) $T(n) = 3T(n/2) + n^2$
(b) $T(n) = 4T(n/2) + n^2$
(c) $T(n) = T(n/2) + 2^n$
(d) $T(n) = 16T(n/4) + n$
(e) $T(n) = 2T(n/2) + n \log n$
(f) $T(n) = 2T(n/2) + n/\log n$
(g) $T(n) = 2T(n/4) + n^{0.51}$
(h) $T(n) = 0.5T(n/2) + 1/n$
(i) $T(n) = 16T(n/4) + n!$
(j) $T(n) = \sqrt{2}T(n/2) + \log n$
(k) $T(n) = 4T(n/2) + cn$
(l) $T(n) = 3T(n/3) + n/2$
(m) $T(n) = 6T(n/3) + n^2 \log n$
(n) $T(n) = 7T(n/3) + n^2$
(o) $T(n) = 4T(n/2) + \log n$
(p) $T(n) = T(n/2) + n(2 - \cos n)$

Solution:

(a) $T(n) = 3T(n/2) + n^2 \implies \Theta(n^2)$ (Case 3)
(b) $T(n) = 4T(n/2) + n^2 \implies \Theta(n^2 \log n)$ (Case 2)
(c) $T(n) = T(n/2) + 2^n \implies \Theta(2^n)$ (Case 3)
(d) $T(n) = 16T(n/4) + n \implies \Theta(n^2)$ (Case 1)
(e) $T(n) = 2T(n/2) + n \log n \implies \Theta(n \log^2 n)$ (Case 2)
(f) $T(n) = 2T(n/2) + n/\log n \implies$ Does not apply (non-poly diff between $f(n)$ and $n^{\log_b a}$)
(g) $T(n) = 2T(n/4) + n^{0.51} \implies \Theta(n^{0.51})$ (Case 3)
(h) $T(n) = 0.5T(n/2) + 1/n \implies$ Does not apply ($a < 1$)
(i) $T(n) = 16T(n/4) + n! \implies \Theta(n!)$ (Case 3)
(j) $T(n) = \sqrt{2}T(n/2) + \log n \implies \Theta(\sqrt{n})$ (Case 1)
(k) $T(n) = 4T(n/2) + cn \implies \Theta(n^2)$ (Case 1)
(l) $T(n) = 3T(n/3) + n/2 \implies \Theta(n \log n)$ (Case 2)
(m) $T(n) = 6T(n/3) + n^2 \log n \implies \Theta(n^2 \log n)$ (Case 3)
(n) $T(n) = 7T(n/3) + n^2 \implies \Theta(n^2)$ (Case 3)
(o) $T(n) = 4T(n/2) + \log n \implies \Theta(n^2)$ (Case 1)
(p) $T(n) = T(n/2) + n(2 - \cos n) \implies$ Does not apply (we are in case 3 but regularity condition violated, e.g., $n = 2\pi k$ for k odd and arbitrarily large: for any such choice of n we have $c \geq 3/2$, violation)

4. Show, using the guess-substitute-verify method, that the solution of $T(n) = T(\lceil n/2 \rceil) + 1$ is $\mathcal{O}(\log n)$. You must not simply ignore the ceiling $\lceil \bullet \rceil$!

Solution:

We guess that $T(n) \leq c \log n$ for some constant $c > 0$. We get (assuming $\log_2$)

$$\begin{aligned}
T(n) &= T(\lceil n/2 \rceil) + 1 \\
&\leq c \log \lceil n/2 \rceil + 1 && (1) \\
&\leq c \log \frac{n+1}{2} + 1 && (2) \\
&\leq c \log \frac{(4/3)n}{2} + 1 && (3) \\
&= c \log \frac{2n}{3} + 1 && (4) \\
&= c(\log(2) + \log(n) - \log(3)) + 1 && (5) \\
&= c(1 + \log(n) - \log(3)) + 1 && (6) \\
&= c \log(n) - (c \log(3) - c - 1) && (7)
\end{aligned}$$

We now need to show that we can find a positive c such that $c \log(3) - c - 1 \geq 0$. This is equivalent to $c \geq 1/(\log(3) - 1) \approx 1.71$.

There are a few minor "tricks" in the above. To get from (1) to (2) we have just replaced $\lceil n/2 \rceil$ with $(n+1)/2$ to get rid of the ceiling. However, the logarithm of n plus something can be a bit awkward, thus in the step from (2) to (3) we have upper bounded the $n+1$ by $(4/3)n$ which is certainly true for n large enough. The rest is just simple algebraic manipulation so that in the end we have a term $c \log n$ plus or minus something (the $c \log n$ of course being our guess).

5. Think about how one may prove the Master theorem. Start with trying to come up with a high-level approach, and then stop there if you wish. Should you want to take this further you may be interested in

   https://www.cs.cornell.edu/courses/cs3110/2013sp/recitations/mm-proof.pdf,

   a document containing a complete proof for the variant with simplified case (2), that is, without the $\log^k n$ term.

Solution:

That document, basically.