Alt-HW1 Solution, CS430, Spring 2016

Instructor: Professor Edward M. Reingold
TA: Jing Zhao, Junze Han & Taeho Jung

January 22, 2016

1

Base Case Proof

$T(3) = 9 = 3^2$. Therefore, when $n = 3$, $T(n) = n^2$.

Inductive Hypothesis

If $a = 3^b$ for $b > 1$, we assume $T(a) = T(3^b) = a^2 = 9^b$.

Inductive Step

The next inductive step to be proved is: if the above hypothesis holds, the following is true

$$\text{If } a = 3^{b+1} \text{ for } b > 1,\ \ T(a) = T(3^{b+1}) = a^2 = 9^{b+1},$$

Proof

$$\begin{aligned}
T(3^{b+1}) &= 6T(3^b) + \frac{1}{3} \cdot (3^{b+1})^2 \quad \text{recurrence relation} \\
&= 6 \cdot 9^b + \frac{1}{3} \cdot (3^{b+1})^2 \quad \text{inductive hypothesis} \\
&= 6 \cdot 9^b + 3 \cdot 9^b = 9^{b+1} \quad \text{simple manipulation} \qquad \blacksquare
\end{aligned}$$

Conclusion

Combining the base case, hypothesis and the inductive step, we are able to conclude $T(n) = n^2$ (where $T(n)$ is recursively defined as above) when $n = 3^k$ for $k > 1$.

2

Let $T(n)$ be the running time needed to perform the binary search in a sorted array $A[1 \cdots n]$. Then, we have:

$$T(n) = \begin{cases} O(1) & n = 1 \\ T(\frac{n}{2}) + O(1) & n > 1 \end{cases}$$

because we always discard a half of the given array at each recurrence.

3

Initialize an output matrix $A'_{m\times o}$ to all zeroes;
for $i = 1 \to m$ **do**
  for $j = 1 \to o$ **do**
    for $k = 1 \to n$ **do**
      $A'_{m\times o}[i][j] = A'_{m\times o}[i][j] + A_{m\times n}[i][k] \cdot A_{n\times o}[k][j]$;
    end
  end
end

Algorithm 1: Multiplication $A_{m\times n} \times A_{n\times o}$

If we assume the time complexities of an addition and a multiplication are both $O(1)$, we can conclude the time complexity is $\Theta(nmo)$ due to the three for-loops.

4

The functions sorted according to their growth rates (non-increasing order):

$$n^n \qquad (n+1)! \qquad \sqrt{n}(n/e)^n \qquad 2^{\lg n} \qquad \sqrt{2}^{\lg n} \qquad \lg n^2$$

Justifications are shown below.

- n^n is $\omega((n+1)!)$ because

$$\lim_{n\to+\infty} \frac{n^n}{(n+1)!} = \lim_{n\to+\infty} \frac{n^n}{\sqrt{2\pi(n+1)}(\frac{n+1}{e})^{n+1}} = \lim_{n\to+\infty} \frac{e^{n+1}}{\sqrt{2\pi(n+1)}(n+1)(1+\frac{1}{n})^n} = +\infty$$

  Therefore, $n^n = \Omega((n+1)!)$

- $(n+1)! = \omega(\sqrt{n}(n/e)^n)$ because

$$(n+1)! \approx \sqrt{2\pi(n+1)}\left(\frac{n+1}{e}\right)^{n+1} = \Theta\left(\sqrt{n+1}(n+1)\left(\frac{n+1}{e}\right)^n\right) = \omega(\sqrt{n}(n/e)^n)$$

  Therefore, $(n+1)! = \Omega(\sqrt{n}(n/e)^n)$ as well.

- $2^{\lg n} = n^{\lg 2} = n^2 \Rightarrow \sqrt{n}(n/e)^n = \Theta(n!) = \omega(2^{\lg n}) \Rightarrow \sqrt{n}(n/e)^n = \Omega(2^{\lg n})$

- For the rest, it is obvious to see the ranks because $\sqrt{2}^{\lg n} = n^{\lg\sqrt{2}} = n^{0.5}$ and $\lg n^2 = 2\lg n$.

5

Using Master Theorem

$$a = 2, b = 2, f(n) = n \Rightarrow \frac{af(n/b)}{f(n)} = \frac{2(n/2)}{n} = 1$$

Therefore, $T(n) = \Theta(f(n)\log_b n) = \Theta(n \log n)$

Using secondary recurrence

Let $n_i = n$ and $n_{i-1} = n/2$. Further, we assume $T(1)$ is the base case and $n_0 = 1$. This does not affect the final result since we are solving for the Θ notation of the function. Then,

$$n_i = 2n_{i-1} \Rightarrow n_i = \alpha 2^i \quad \text{(corresponds to } (E-2))$$

Since $n_0 = 1$, $\alpha = 1$ and $n_i = 2^i$. Further, define $F(i) = T(n_i)$. Then, the original recurrence:

$$T(n) = T(n_i) = 2T(n/2) + n = 2T(n_{i-1}) + n$$

becomes

$$F(i) = 2F(i-1) + n$$

We have supposed $n = n_i$, and we derived that $n_i = 2^i$. Therefore, the final recurrence to solve is:

$$F(i) = 2F(i-1) + 2^i$$

which is annihilated by $(E-2)^2$. The corresponding closed formula is $(\alpha_1 i + \alpha_2)2^i$, which is $\Theta(i2^i)$. Recall that $n = 2^i$. We can achieve the final Θ notation by undoing the substitution as follows:

$$T(n) = F(i) = \Theta(i2^i) = \Theta(\log_2 n \cdot 2^{\log_2 n}) = \Theta(n \log n)$$