Advanced Analysis

Kuan-Yu Chen (陳冠宇)

2020/09/23 @ TR-212, NTUST

Review

- Please check the uploaded HW file on the moodle!

- Space and Time complexity
  - Big-Oh
  - Omega
  - Theta

Definition [*Big "oh"*]: $f(n) = O(g(n))$ (read as "f of n is big oh of g of n") iff (if and only if) there exist positive constants c and n_0 such that $f(n) \le cg(n)$ for all n, $n \ge n_0$. □

Definition: [Omega] $f(n) = \Omega(g(n))$ (read as "f of n is omega of g of n") iff there exist positive constants c and n_0 such that $f(n) \ge cg(n)$ for all n, $n \ge n_0$. □

Definition: [Theta] $f(n) = \Theta(g(n))$ (read as "f of n is theta of g of n") iff there exist positive constants c_1, c_2, and n_0 such that $c_1 g(n) \le f(n) \le c_2 g(n)$ for all n, $n \ge n_0$. □

Little-Oh

$$o(g(n)) = \{f(n) : \text{ for any positive constant } c > 0, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \le f(n) < cg(n) \text{ for all } n \ge n_0\}.$$

- The value of n_0 may depend on c

- The definitions of O-notation and o-notation are similar

Definition [*Big "oh"*]: $f(n) = O(g(n))$ (read as "f of n is big oh of g of n") iff (if and only if) there exist positive constants c and n_0 such that $f(n) \le cg(n)$ for all n, $n \ge n_0$. □

  - $f(n) = O(g(n))$, the bound $0 \le f(n) \le cg(n)$ holds for ***some*** constant $c > 0$
  - $f(n) = o(g(n))$, the bound $0 \le f(n) < cg(n)$ holds for ***all*** constants $c > 0$

- Examples:
  - $2n = o(n^2)$
  - $2n^2 \ne o(n^2)$

Little-Omega

$$\omega(g(n)) = \{f(n) : \text{ for any positive constant } c > 0, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \le cg(n) < f(n) \text{ for all } n \ge n_0\} .$$

- By analogy, ω-notation is to Ω-notation as o-notation is to O-notation

Definition: [Omega] $f(n) = \Omega(g(n))$ (read as "f of n is omega of g of n") iff there exist positive constants c and n_0 such that $f(n) \ge cg(n)$ for all n, $n \ge n_0$. □

- Examples:
  - $\frac{n^2}{2} = \omega(n)$
  - $\frac{n^2}{2} \neq \omega(n^2)$

Summary

$$
\begin{array}{lcl}
f(n) = \Theta(g(n)) \text{ and } g(n) = \Theta(h(n)) & \text{imply} & f(n) = \Theta(h(n)) \\
f(n) = O(g(n)) \text{ and } g(n) = O(h(n)) & \text{imply} & f(n) = O(h(n)) \\
f(n) = \Omega(g(n)) \text{ and } g(n) = \Omega(h(n)) & \text{imply} & f(n) = \Omega(h(n)) \\
f(n) = o(g(n)) \text{ and } g(n) = o(h(n)) & \text{imply} & f(n) = o(h(n)) \\
f(n) = \omega(g(n)) \text{ and } g(n) = \omega(h(n)) & \text{imply} & f(n) = \omega(h(n))
\end{array}
$$

$$
\begin{aligned}
f(n) &= \Theta(f(n)) \\
f(n) &= O(f(n)) \\
f(n) &= \Omega(f(n))
\end{aligned}
$$

$$f(n) = \Theta(g(n)) \text{ if and only if } g(n) = \Theta(f(n))$$

$$
\begin{aligned}
f(n) = O(g(n)) &\text{ if and only if } g(n) = \Omega(f(n)) \\
f(n) = o(g(n)) &\text{ if and only if } g(n) = \omega(f(n))
\end{aligned}
$$

Master Method.

- The master method provides a "cookbook" method for solving recurrences of the form

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

where $a \geq 1$ and $b > 1$ are constants and $f(n)$ is a positive function

1. If $f(n) = O(n^{\log_b a-\epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$
3. If $f(n) = \Omega(n^{\log_b a+\epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

Master Method..

- The master method theorem

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

1. If $f(n) = O(n^{\log_b a-\epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$
3. If $f(n) = \Omega(n^{\log_b a+\epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

  – Take $T(n) = 9T\left(\frac{n}{3}\right) + n$ for example
    - $a = 9, b = 3$, and $f(n) = n \Longrightarrow n^{\log_b a} = n^{\log_3 9} = n^2$
    - $f(n) = n = O\left(n^{\log_b a-\epsilon}\right) = O\left(n^{\log_3 9-\epsilon}\right)$, where $\epsilon = 1$
    - $T(n) = \Theta\left(n^{\log_b a}\right) = \Theta\left(n^{\log_3 9}\right) = \Theta\left(n^2\right)$
    - Case1

Master Method...

- The master method theorem

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

1. If $f(n) = O(n^{\log_b a-\epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$
3. If $f(n) = \Omega(n^{\log_b a+\epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

– Take $T(n) = T\left(\frac{2n}{3}\right) + 1$ for example

- $a = 1, b = \frac{3}{2}$, and $f(n) = 1 \implies n^{\log_b a} = n^{\log_{\frac{3}{2}} 1} = n^0 = 1$
- $f(n) = 1 = \Theta(1) = \Theta\left(n^{\log_b a}\right)$
- $T(n) = \Theta\left(n^{\log_b a} \log_2 n\right) = \Theta(\log_2 n)$
- Case2

Master Method....

- The master method theorem

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

1. If $f(n) = O(n^{\log_b a-\epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$
3. If $f(n) = \Omega(n^{\log_b a+\epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

- Take $T(n) = 3T\left(\frac{n}{4}\right) + n\log_2 n$ for example
  - $a = 3, b = 4$, and $f(n) = n\log_2 n \Longrightarrow n^{\log_b a} = n^{\log_4 3}$
  - $f(n) = \Omega(n^{\log_4 3+\epsilon})$
  - $af\left(\frac{n}{b}\right) = 3f\left(\frac{n}{4}\right) = 3\frac{n}{4}\log_2\frac{n}{4} = 3\frac{n}{4}(\log_2 n - \log_2 4)$
    $= \frac{3}{4}n\log_2 n - \frac{3}{2}n \leq cn\log_2 n = cf(n)$, when $c = \frac{3}{4}$
  - $T(n) = \Theta\big(f(n)\big) = \Theta(n\log_2 n)$
  - Case3

Master Method.....

- The master method theorem

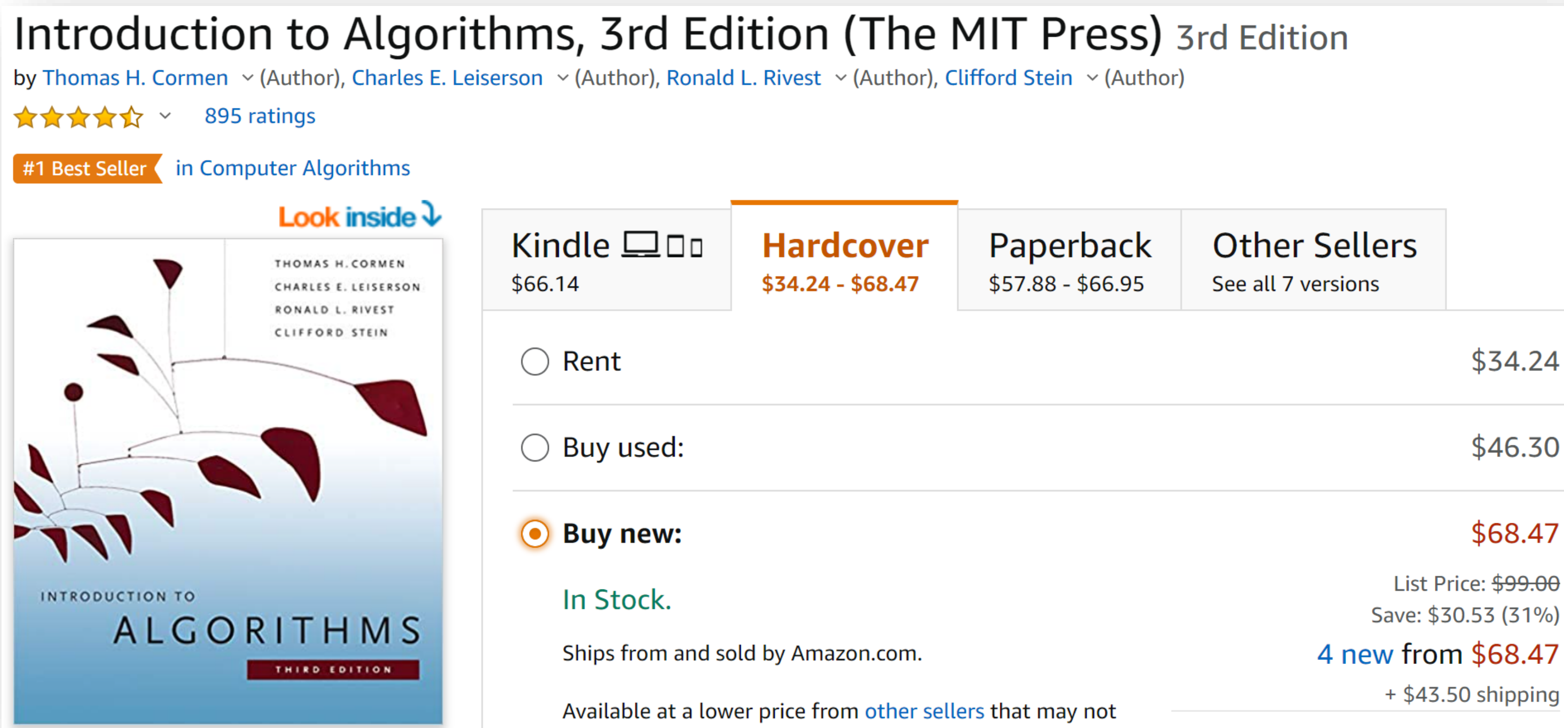
$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

1. If $f(n) = O(n^{\log_b a-\epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$
3. If $f(n) = \Omega(n^{\log_b a+\epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

   – In each of the three cases, we compare $f(n)$ with $n^{\log_b a}$
   – In case1, $n^{\log_b a}$ is larger than $f(n)$, thus $T(n) = \Theta(n^{\log_b a})$
   – In case3, $f(n)$ is larger than $n^{\log_b a}$, thus $T(n) = \Theta(f(n))$
   – In case2, $f(n)$ and $n^{\log_b a}$ are the same size, we multiply by a logarithmic factor, and the solution is $T(n) = \Theta(n^{\log_b a} \log_2 n) = \Theta(f(n) \log_2 n)$

Master Method……

- Check out the proof from:



Questions?

kychen@mail.ntust.edu.tw