CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT (Data Structures & Algorithms)

L/O/G/O

GV: HUỲNH THỊ THANH THƯƠNG
Email: thuonghtt@uit.edu.vn

HW#03: Big-O notation

❖ Bài tập 1:

a). Hãy cho biết ý nghĩa của "độ phức tạp" khi đề cập đến thuật toán?

b) Hãy cho biết ý kiến của bạn về nhận định dưới đây và giải thích vì sao?

"Khi nghiên cứu về các thuật toán, người ta quan tâm đặc biệt đến tính hiệu quả về thời gian của chúng nhưng thường là quan tâm đến bậc tăng trưởng (order of growth) của hàm thời gian thực hiện của thuật toán, chứ không phải là bản thân thời gian thực hiện T(n)".

c) Nói về ĐPT tức là đề cập tới các ký hiệu tiệm cận, mà có nhiều ký hiệu khác nhau. Vậy khi nào(trong trường hợp nào) thì nên dùng ký hiệu nào?

PS: viết ngắn gọn nhưng đủ ý, viết dài dòng lê thê cô đọc mệt

1-1 Comparison of running times

For each function $f(n)$ and time t in the following table, determine the largest size n of a problem that can be solved in time t, assuming that the algorithm to solve the problem takes $f(n)$ microseconds.

Bài tập 2

Lưu ý: cần giải thích cách làm (cách tính toán) chứ không phải chỉ điền số vào bảng. Không cần giải thích hết các hàm, chỉ cần lấy ít nhất 3 hàm nào đó làm ví dụ . Nếu viết code thì đưa code vào để giải thích cũng được

Notes for Chapter 1 15

	1 second	1 minute	1 hour	1 day	1 month	1 year	1 century
$\lg n$							
$\sqrt{n}$							
n							
$n \lg n$							
n^2							
n^3							
2^n							
$n!$							

HW#03: Big-O notation

- Bài tập 3:

a) Phép suy ra bên dưới là đúng hay sai và vì sao?

$$\frac{1}{2}n^2 = O(n^2)$$

$$n^2 + 1 = O(n^2)$$

$$\Rightarrow \frac{1}{2}n^2 = n^2 + 1 \quad ???$$

HW#03: Big-O notation

❖ **Bài tập 3**:

b) Chứng minh

$n^3 \notin O(n^2)$

$n^4 + n + 1 \notin O(n^2)$

$O(n^2) \neq O(n)$

c) Chứng minh:

Nếu $T(n) = a_k n^k + ... + a_1 n + a_0$ thì

$T(n) = O(n^k)$

trong đó, các a_i với $i = 0, 1, ..., k$ là các hằng số thực.

HW#03: Big-O notation

- **Bài tập 4**: Với mỗi nhóm hàm bên dưới, hãy sắp xếp tăng dần "theo Big-O nhỏ nhất", có giải thích ngắn gọn cách so sánh

Group 1:

$$f_1(n) = \binom{n}{100}$$
$$f_2(n) = n^{100}$$
$$f_3(n) = 1/n$$
$$f_4(n) = 10^{1000}n$$
$$f_5(n) = nlogn$$

Group 2:

$$f_1(n) = 2^{2^{1000000}}$$
$$f_2(n) = 2^{100000n}$$
$$f_3(n) = \binom{n}{2}$$
$$f_4(n) = n\sqrt{n}$$

Group 3:

$$f_1(n) = n^{\sqrt{n}}$$
$$f_2(n) = 2^n$$
$$f_3(n) = n^{10} \cdot 2^{n/2}$$
$$f_4(n) = \sum_{i=1}^{n}(i+1)$$

Ký hiệu: log là log cơ số 2, $\binom{n}{k}$ là tổ hợp chập k của n

HW#03: Big-O notation

Group 4:

$f_1(n) = n^4\binom{n}{2}$

$f_2(n) = \sqrt{n}(\log n)^4$

$f_3(n) = n^{5\log n}$

$f_4(n) = 4\log n + \log\log n$

$f_5(n) = \sum_{i=1}^{n} i$

Group 5:

$f_6(n) = n^{\sqrt{n}}$

$f_7(n) = n^{\log n}$

$f_8(n) = 2^{n/2}$

$f_9(n) = 3^{\sqrt{n}}$

$f_{10}(n) = 4^{n^{1/4}}$

Group 6:

$f_1(n) = n^{0.999999}\log n$

$f_2(n) = 10000000n$

$f_3(n) = 1.000001^n$

$f_4(n) = n^2$

Group 7: $(n-2)!,\ 5\lg(n+100)^{10},\ 2^{2n},\ 0.001n^4 + 3n^3 + 1,\ \ln^2 n,\ \sqrt[3]{n},\ 3^n$

HW#03: Các ký hiệu tiệm cận khác

Bài tập 5: Chứng minh

a. O(C) = O(1) với C là hằng số

b. Nếu f(n) ∈ O(g(n)) và g(n) ∈ O(h(n))
thì f(n) ∈ O(h(n))

c. $\max\{f(n), g(n)\} = \Theta(f(n) + g(n))$

d. If $t(n) \in O(g(n))$, then $g(n) \in \Omega(t(n))$.

e. $\Theta(\alpha g(n)) = \Theta(g(n))$, where $\alpha > 0$.

f. $\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$.

HW#03: Các ký hiệu tiệm cận khác

- **Bài tập 6: Các khẳng định bên dưới là đúng hay sai? Vì sao?**
  a. Nếu $f(n) = \Theta(g(n))$ và $g(n) = \Theta(h(n))$, thì $h(n) = \Theta(f(n))$
  b. Nếu $f(n) = O(g(n))$ và $g(n) = O(h(n))$, thì $h(n) = \Omega(f(n))$
  c. Nếu $f(n) = O(g(n))$ và $g(n) = O(f(n))$, thì $f(n) = g(n)$
  d. $n/100 = \Omega(n)$
  e. $f(n) + O(f(n)) = \Theta(f(n))$
  f. $2^{10n} = O(2^n)$
  g. $2^{n+10} = O(2^n)$
  h. $\log_{10} n = \Theta(\log_2 n)$

Bài tập 7: Ước lượng nhanh độ phức tạp của giải thuật đệ quy dùng Định lý Master

- SV tự nghiên cứu về Định lý Master (theo các slide sau) và làm bài tập đính kèm

Master Theorem (1): Dạng đơn giản

❖ Định lý Master cho phép ước lượng nghiệm của các phương trình đệ quy có dạng (*đơn giản*):

$$\begin{aligned} T(n) &= aT(\tfrac{n}{b}) + f(n) \\ T(1) &= c \end{aligned}$$

where $a \geq 1, b \geq 2, c > 0$. *If* $f(n) \in \Theta(n^d)$ *where* $d \geq 0$, *then*

$$T(n) = \begin{cases} \Theta(n^d) & \text{if } a < b^d \\ \Theta(n^d \log n) & \text{if } a = b^d \\ \Theta(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

Master Theorem (1)

❖ Ví dụ:

$$T(n) = T\left(\frac{n}{2}\right) + \frac{1}{2}n^2 + n$$

$$T(n) = \begin{cases} \Theta(n^d) & \text{if } a < b^d \\ \Theta(n^d \log n) & \text{if } a = b^d \\ \Theta(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

a = 1, b = 2, d = 2

→ Vì $1 < 2^2$ TH1 được áp dụng

$$T(n) \in \Theta\left(n^d\right) = \Theta(n^2)$$

Master Theorem (1)

❖ Ví dụ:

$$T(n) = 2T\left(\frac{n}{4}\right) + \sqrt{(n)} + 42$$

$$T(n) = \begin{cases} \Theta(n^d) & \text{if } a < b^d \\ \Theta(n^d \log n) & \text{if } a = b^d \\ \Theta(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

a = 2, b = 4, d = 1/2

→ Vì $2 = 4^{1/2}$ TH2 được áp dụng

$$T(n) \in \Theta\left(n^d log n\right) = \Theta\left(\sqrt{(n)} log n\right)$$

Master Theorem: Ưu nhược điểm

❖ Nhược điểm:

- Chỉ áp dụng cho phương trình đệ quy có dạng như trên
- Không thể dùng định lý Master (dạng 1) nếu:

- T(n) không đơn điệu. VD: $T(n) = sin(n)$
- f(n) không phải hàm đa thức. VD: $T(n) = 2T\left(\frac{n}{2}\right) + 2^n$
- b không thể biểu diễn như 1 hằng số.

Master Theorem (2): Dạng tổng quát

❖ The Master Theorem applies to recurrence of the following form (general - tổng quát hơn 1):

$$T(n) = aT(n/b) + f(n)$$

where $a \geq 1$ and $b > 1$ are constants

$f(n)$ is an asymptotically positive function

Master Theorem (2)

- Có 3 trường hợp:

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 $f(n) = \Theta(n^{\log_b a} \log^k n)$ with $k \geq 0$,
   then $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$

3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ with $\epsilon > 0$,
   and $f(n)$ satisfies the regularity condition, then $T(n) = \Theta(f(n))$

Regularity condition:

$af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n.

Master Theorem (2)

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 $f(n) = \Theta(n^{\log_b a} \log^k n)$ with $k \geq 0$, then $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ with $\epsilon > 0$, and $f(n)$ satisfies the regularity condition, then $T(n) = \Theta(f(n))$

❖ Ví dụ: $T(n) = 2T\left(\frac{n}{2}\right) + nlogn$

a = 2, b = 2, f(n) không phải là đa thức

→Không áp dụng Master Theorem (1) được, tuy nhiên: $f(n) \in \Theta(nlogn)$

→ với k = 1, áp dụng (2.2) ta được

$$T(n) \in \Theta(nlog^2 n)$$

Master Theorem (2)

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 $f(n) = \Theta(n^{\log_b a} \log^k n)$ with $k \geq 0$,
   then $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ with $\epsilon > 0$,
   and $f(n)$ satisfies the regularity condition, then $T(n) = \Theta(f(n))$

❖ Ví dụ: $T(n) = 4T\left(\frac{n}{2}\right) + n^3$

a = 4, b = 2, và $n^{log_b a} = n^{log_2 4} = n^2$

Ta thấy, $f(n) = n^3 = \Omega(n^{2+\epsilon}), \epsilon = 0.5$

và $af(n/b) \leq cf(n)$

(vì $4\left(\frac{n}{2}\right)^3 = \frac{n^3}{2} \leq cn^3, c = 1/2$)

→áp dụng (2.3) ta được

$$T(n) = \Theta\big(f(n)\big) = \Theta(n^3)$$

❖Giải bằng định lý Master. Ghi rõ Áp dụng trường hợp nào (Case 1, Case 2, Case 3) của Định lý số mấy (Dạng đơn giản – 1 hay Dạng tổng quát – 2) ? Câu nào không áp dụng được định lý Master thì giải thích vì sao?

1. $T(n) = 3T\left(\frac{n}{2}\right) + n^2$

2. $T(n) = 7T\left(\frac{n}{3}\right) + n^2$

3. $T(n) = 3T\left(\frac{n}{3}\right) + \frac{n}{2}$

4. $T(n) = 16T\left(\frac{n}{4}\right) + n$

5. $T(n) = 2T\left(\frac{n}{4}\right) + n^{0.51}$

6. $T(n) = 3T\left(\frac{n}{2}\right) + n$

7. $T(n) = 3T\left(\frac{n}{3}\right) + \sqrt{(n)}$

8. $T(n) = 4T\left(\frac{n}{2}\right) + cn$

9. $T(n) = 4T\left(\frac{n}{4}\right) + 5n$

10. $T(n) = 5T\left(\frac{n}{4}\right) + 4n$

❖Giải bằng định lý Master. Ghi rõ Áp dụng trường hợp nào (Case 1, Case 2, Case 3) của Định lý số mấy (Dạng đơn giản – 1 hay Dạng tổng quát – 2) ? Câu nào không áp dụng được định lý Master thì giải thích vì sao?

11. $T(n) = 4T\left(\frac{n}{5}\right) + 5n$

12. $T(n) = 25T\left(\frac{n}{5}\right) + n^2$

13. $T(n) = 10T\left(\frac{n}{3}\right) + 17n^{1.2}$

14. $T(n) = 7T\left(\frac{n}{2}\right) + n^3$

15. $T(n) = 4T\left(\frac{n}{2}\right) + logn$

16. $T(n) = 4T\left(\frac{n}{5}\right) + logn$

17. $T(n) = \sqrt{(2)}T\left(\frac{n}{2}\right) + logn$

18. $T(n) = 2T\left(\frac{n}{3}\right) + nlogn$

19. $T(n) = 3T\left(\frac{n}{4}\right) + nlogn$

20. $T(n) = 6T\left(\frac{n}{3}\right) + n^2logn$

❖Giải bằng định lý Master. Ghi rõ Áp dụng trường hợp nào (Case 1, Case 2, Case 3) của Định lý số mấy (Dạng đơn giản – 1 hay Dạng tổng quát – 2) ? Câu nào không áp dụng được định lý Master thì giải thích vì sao?

21. $T(n) = 3T\left(\frac{n}{5}\right) + log^2 n$

22. $T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{logn}$

23. $T(n) = 2^n T\left(\frac{n}{2}\right) + n^n$

24. $T(n) = 0.5T\left(\frac{n}{2}\right) + n$

25. $T(n) = T\left(\frac{n}{2}\right) + n(2 - cosn)$

26. $T(n) = 64T\left(\frac{n}{8}\right) - n^2 logn$

27. $T(n) = T\left(\frac{n}{2}\right) + 2^n$

28. $T(n) = 16T\left(\frac{n}{4}\right) + n!$

HW#03: Big-O notation

- **Bài tập 8 (bonus-không bắt buộc): chứng minh các tính chất sau:**
  - $n + n^2 O(\ln n) = O(n^2 \ln n)$
  - $g(n) \in O(h(n)) \Rightarrow O(g(n)) \subseteq O(h(n))$
  - $O(f(n)) = O(g(n)) \Leftrightarrow g(n) \in O(f(n))$ và $f(n) \in O(g(n))$
  - $O(f(n) \subset O(g(n)) \Leftrightarrow f(n) \in O(g(n))$ và $g(n) \notin O(f(n))$
  - $f(n) \in O(n) \Rightarrow \quad 2^{f(n)} \in O(2^n)$

Gợi ý

$f(n) = n^3 + O(n^2)$
means
$f(n) = n^3 + h(n)$
for some $h(n) \in O(n^2)$

$n^2 + O(n) = O(n^2)$
means
for any $f(n) \in O(n)$:
$n^2 + f(n) = h(n)$
for some $h(n) \in O(n^2)$

HW#03: Các ký hiệu tiệm cận khác

- **Bài tập 9 (bonus-không bắt buộc): Chứng minh theo 2 cách là dùng giới hạn và định nghĩa**

SV xem hướng dẫn trong 3 slide sau về những ký hiệu mới và dùng lim để kết luận quan hệ

- Cho $f(n) = \sum_{i=1}^{n} i$ và g(n) = n^2. Chứng minh f(n) = Θ (g(n))
- Chứng minh: $\frac{1}{2}n^2 - 3n = \Theta(n^2)$
- Chứng minh: nlogn – 2n + 13 = Ω (nlogn)
- Chứng minh: $\log_3(n^2) = \Theta \log_2(n^3)$
- Chứng minh: $n^{\lg 4} \in \omega (3^{\lg n})$
- Chứng minh: $\lg^2 n \in o(n^{1/2})$
- Chứng minh: $\frac{n^2}{2} \neq \omega (n^2)$

o-notation and ω-notation

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

EXAMPLE: $2n^2 = o(n^3)$ $(n_0 = 2/c)$

$$\lim_{n \to \infty} [f(n) / g(n)] = 0$$

o-notation and ω-notation

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

EXAMPLE: $\sqrt{n} = \omega(\lg n)$ $\quad (n_0 = 1+1/c)$

$$\lim_{n \to \infty} [f(n) / g(n)] = \infty$$

Sử dụng các giới hạn để suy ra quan hệ

Limits

- $\lim_{n\to\infty} [f(n) / g(n)] = 0 \Rightarrow f(n) \in o(g(n))$
- $\lim_{n\to\infty} [f(n) / g(n)] < \infty \Rightarrow f(n) \in O(g(n))$
- $0 < \lim_{n\to\infty} [f(n) / g(n)] < \infty \Rightarrow f(n) \in \Theta(g(n))$
- $0 < \lim_{n\to\infty} [f(n) / g(n)] \Rightarrow f(n) \in \Omega(g(n))$
- $\lim_{n\to\infty} [f(n) / g(n)] = \infty \Rightarrow f(n) \in \omega(g(n))$
- $\lim_{n\to\infty} [f(n) / g(n)]$ undefined $\Rightarrow$ can't say