Nhập môn
Phân tích độ phức tạp thuật toán

Chủ đề 2: Những khái niệm và vấn đề trong phân tích độ phức tạp thuật toán

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Tóm tắt nội dung chủ đề 2

Chủ đề này giới thiệu về

- Các khái niệm cơ bản về độ phức tạp thuật toán
- Khái niệm về việc phân tích độ phức tạp
- Các tiếp cận để ước lượng và phân tích độ phức tạp thuật toán

Nội dung chủ đề 2

- Bài toán & kích thước, Thuật toán, Độ phức tạp
- Phân loại độ lớn của các hàm
  - Các ký hiệu O lớn ("Big – Oh") và những ký hiệu khác "Big - Omega", "Big -théta"
  - Kỹ thuật toán học bổ trợ
- Khái niệm về đánh giá độ phức tạp
- Các tiếp cận và phương pháp đánh giá độ phức tạp thuật toán
  - Phương pháp thực nghiệm
  - Phương pháp lý thuyết và vài công cụ toán học

Bài toán, Thuật toán, Độ phức tạp

- Mỗi bài toán có nhiều cách giải
  - Dùng giấy và bút với một khối lượng tính toán không quá nhiều
  - Một cách thức để có thể "*giải tự động*"
  - → Thuật toán, Thuật giải, Giải thuật…
  - → Giải nhờ chương trình chạy trên máy tính
- Mỗi bài toán có thể giải nhờ nhiều thuật toán khác nhau

→ Thuật toán nào "*tốt hơn*" ?

→ Tính "*phức tạp*" của mỗi thuật toán…

Độ phức tạp thuật toán

- Độ phức tạp về thời gian
  - Liên quan đến việc chạy nhanh hay chậm, mang tính ước lượng gần đúng, không quá chi li lặt vặt
  - Không nên quá phụ thuộc vào các yếu tố phần cứng

  → Mô hình nào, đặc trưng gì có thể dùng cho độ phức tạp thời gian?
- Độ phức tạp về không gian: đặc trưng về tiêu thụ tài nguyên, ví dụ như bộ nhớ…

→ Môn học quan tâm chủ yếu độ phức tạp thời gian

→ Thuật ngữ "*độ phức tạp*" = "*độ phức tạp thời gian*"

Kích thước bài toán và độ phức tạp

- Kích thước bài toán P
  - Một con số n nào đó (qui về nguyên dương)
  - Khi n càng lớn thì việc giải bài toán càng chậm (với tất cả các thuật toán đã biết được dùng để giải P)
  - Cộng đồng (nghiên cứu hay nghề nghiệp) hiểu về kích thước này khi nói về P
- Giả sử thuật toán A giải bài toán P có kích thước n

→ Một hàm $f_{P,A}(n)$ đặc trưng cho thời gian chạy

→ Thường gọi tắt là $f(n)$ (khi ngầm hiểu P và A)

Độ phức tạp và độ lớn của hàm

- Thuật toán **A** giải bài toán **P**, kích thước ***n***
- Hàm $f(n)$ đặc trưng thời gian chạy của thuật toán **A**.
- "Độ lớn" của $f(n)$ sẽ được dùng để đặc trưng hóa độ phức tạp của thuật toán **A**.
- → Sử dụng đại lượng nào liên quan đến $f(n)$ để đặc trưng cho độ lớn của hàm?
- → Và có thể so sánh được độ lớn các hàm…
- Các khái niệm toán học: O lớn ("Big – Oh") và "Big - Omega", "Big - théta", "Small Oh".

Ví dụ minh họa

- Xem minh họa cụ thể trên bảng viết…

So sánh độ lớn các hàm

Cho hai hàm số thực $f(x)$ và $g(x)$:

$$f, g: ? \rightarrow ?$$

- Ta nói "f nhỏ thua g" và ký hiệu f ? g nếu tồn tại hai hằng số dương M và N_0 sao cho:

$$\forall x \in ?, x \geq N_0 \Rightarrow |f(x)| \leq M.|g(x)|$$

- Diễn giải ý nghĩa: "Giá trị tuyệt đối của hàm $f(x)$ nhỏ hơn giá trị tuyệt đối của hàm $g(x)$ sai khác một hệ số tỉ lệ M khi x đủ lớn".

Minh họa về quan hệ � giữa các hàm

- Xem minh họa cụ thể trên bảng viết…

Vài lưu ý về quan hệ độ lớn hàm

- Hai hằng số M và N_0 phải không phụ thuộc vào biến số x.
- Hoàn toàn có thể giả sử M và N_0 nguyên dương (không ảnh hưởng gì hết).
- Quan hệ ⍰ có hai tính chất: phản xạ và bắc cầu, tức là "*quan hệ tiền thứ tự*".
- Khi áp dụng vào thuật toán thì ta chỉ quan tâm đến $f(n)$, $g(n)$ với n nguyên dương.

Khái niệm O lớn, "Big – Oh" (1)

Cho trước hàm số $g(x)$,

Tập hợp hàm $O(g)$ được định nghĩa là:

$$O(g) = \{ f \;⍰\; f : ⍰ \rightarrow ⍰,\, f ⍰ g \}$$

Tức là $O(g)$ gồm các hàm f mà nhỏ thua g.

Vài tính chất của tập $O(g)$:

- $f ⍰ g \Rightarrow O(f) \subseteq O(g)$;
- $f \in O(g) \Rightarrow O(f) \subseteq O(g)$;
- $(f ⍰ g \wedge g ⍰ f) \Leftrightarrow f \in O(g) \wedge g \in O(f)$

  $\Leftrightarrow O(f) = O(g)$

Khái niệm O lớn, "Big – Oh" (2)

Cho hai hàm số $f(x)$, $g(x)$.

Khi $f \in O(g)$ thì ta cũng viết là $f = O(g)$.

Như vậy dấu bằng (dấu =) nói trên thực sự chỉ là dấu thuộc (dấu $\in$).

Ta cần thận trọng để tránh sai lầm.

- Không thể viết $O(g) = f$ (**viết sai !**)
- $f = O(g) \ \wedge \ g = O(h) \ \Rightarrow f = O(h)$
- $f \neq O(g)$ có nghĩa là $f \notin O(g)$
- Khi $f = O(g)$ thì $O(f) \subseteq O(g)$, nhưng chưa chắc $O(f) = O(g)$.

Ví dụ về ký hiệu O lớn

- Xem minh họa cụ thể trên bảng viết…
- Ví dụ $f = O(g)$
- Ví dụ $f = O(g)$ và $g = O(f)$
- Ví dụ $f = O(g)$ và $g \neq O(f)$
- Ví dụ $f \neq O(g)$ và $g \neq O(f)$

Ký hiệu Ω lớn – Big Omega

- Ký hiệu này nghĩa là "trội hơn" thay vì "nhỏ thua"
- Cũng được định nghĩa khi x đủ lớn và sai khác hằng số M.
- Định nghĩa (Knuth): $f = \Omega(g) \Leftrightarrow g = O(f)$

  (lưu ý dấu bằng có ý nghĩa là $\in$)
- Ký hiệu Ω cũng có những định nghĩa toán học khác, không giống trong độ phức tạp

Ví dụ trực quan về độ phức tạp thuật toán

- Xem minh họa cụ thể trên bảng viết…
- Vấn đề thuật ngữ quen dùng…

Quan hệ tương đương giữa các hàm

Cho hai hàm số $f(x)$, $g(x)$.

Ta nói f tương đương g và viết là $f \sim g$ khi

$$f = O(g) \text{ và } g = O(f).$$

- Quan hệ $\sim$ là một quan hệ tương đương (có ba tính chất: phản xạ, đối xứng, bắc cầu)
- $f \sim g \Leftrightarrow (f \in O(g) \wedge g \in O(f)) \Leftrightarrow O(f) = O(g)$
- Như vậy quan hệ $\sim$ chia tập các hàm thành "*các lớp tương đương*"

Ký hiệu Θ lớn – Big théta

Cho trước hàm số $g(x)$,

Tập hợp hàm $\Theta(g)$ được định nghĩa là:

$$\Theta(g) = \{ f \;\; ? \;\; f: ? \rightarrow ?, f \sim g \}$$

Tức là $\Theta(g)$ gồm các hàm f tương với g.

Ta cũng ký hiệu $f = \Theta(g)$ khi $f \in \Theta(g)$.

Vài tính chất của tập $\Theta(g)$:

- $\Theta(f) = \Theta(g) \Leftrightarrow f \sim g \Leftrightarrow f \in \Theta(g) \Leftrightarrow g \in \Theta(f)$
- $f = \Theta(g) \Leftrightarrow g = \Theta(f)$

Vài mệnh đề cho O lớn và Θ lớn

- <u>Mệnh đề</u>: Cho hai hàm số $f(x)$, $g(x)$. Nếu giới hạn

$$\lim_{x\to\infty}\frac{f(x)}{g(x)}$$

tồn tại và hữu hạn thì ta có f=O(g).

- <u>Mệnh đề</u>: Nếu giới hạn nói trên tồn tại và có giá trị là số L $\neq$ 0 thì f = Θ(g).
- Ký hiệu “o nhỏ”: Nếu giới hạn nói trên bằng 0 thì ta nói f=o(g). Điều này có nghĩa là:

$$f=o(g) \Rightarrow f=O(g)$$

(điều đảo lại không đúng).

Ví dụ áp dụng mệnh đề giới hạn cho O lớn và Θ lớn

- Xem trình bày trên bảng…
- Dùng qui tắc L’hopital để tìm giới hạn…
- Kết luận về O lớn, o nhỏ, Θ lớn của một vài hàm thông dụng…

Một số lớp hàm thông dụng

- Logarit $O(\log(n))$; tương đương log: $\Theta(\log(n))$
- Tuyến tính $O(n)$
- Tương đương tuyến tính $\Theta(n)$
- Bậc hai hay bình phương $O(n^2)$
- Tương đương bậc hai $\Theta(n^2)$
- Đa thức $O(n^k)$ với hằng số k nguyên dương
- Tương đương đa thức $\Theta(n^k)$, $k \in$ [illegible]
- Mũ $O(a^n)$ với $a > 1$
- Giai thừa $O(n!)$

Độ phức tạp thuật toán

- Thuật toán **A** giải bài toán **P**, kích thước $\boldsymbol{n}$
- Giả sử hàm $f(n)$ đặc trưng thời gian chạy của thuật toán **A**.
- Nếu $f \in O(g)$ với g là một hàm quen thuộc thì ta nói thuật toán **A** có độ phức tạp $g(n)$.

  Ví dụ: Nếu $f \in O(n)$ thì thuật toán **A** có độ phức tạp tuyến tính hay độ phức tạp $O(n)$.

  Vì $O(n) \subseteq O(n^2)$ nên $f \in O(n^2)$, ta cũng nói thuật toán **A** có độ phức tạp bình phương, nhưng nói vậy sẽ "lỏng" hơn tuyến tính.

Ví dụ về độ phức tạp thuật toán

- Xem minh họa cụ thể trên bảng viết…
- Độ phức tạp theo O lớn …
- Độ phức tạp theo Θ lớn …

Khái niệm về đánh giá độ phức tạp

Thuật toán **A** giải bài toán **P**, kích thước ***n***.

Để đánh giá độ phức tạp thuật toán **A**:

- Bằng cách nào đó tìm ra một hàm $f(n)$ đặc trưng thời gian chạy của **A**.
- Tìm một hàm g quen thuộc sao cho $f \in O(g)$, hàm g thuộc lớp hàm có độ lớn nhỏ nhất có thể được.

Kết luận thuật toán **A** có độ phức tạp $g(n)$.

Phương pháp đánh giá độ phức tạp

- Phương pháp thực nghiệm: chạy thử và thống kê trên dữ liệu nhập tiêu biểu.
- Phương pháp lý thuyết
  - Phép đếm và qui nạp
  - Áp dụng lý thuyết xác suất
  - Áp dụng lý thuyết hoán vị
  - Áp dụng hàm sinh
  - Vài công cụ toán học liên quan thuật toán dùng để bài toán đang xét.

CÂU HỎI

&

THẢO LUẬN