ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

- KHÁI NIỆM
- PHÉP TÍNH
- CÁC BƯỚC PHÂN TÍCH THUẬT TOÁN
- SỰ PHÂN LỚP CÁC THUẬT TOÁN
- PHÂN TÍCH TRƯỜNG HỢP TRUNG BÌNH

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖KHÁI NIỆM

- Độ phức tạp của thuật toán là tốc độ tăng chi phí cho việc thực hiện thuật toán dựa trên tốc độ tăng số lượng giá trị đầu vào. Chi phi cho việc thực hiện thuật toán gồm 2 dạng: chi phí về thời gian và chi phí về không gian.

  + Chi phí về thời gian: Thời gian máy tính thực hiện thuật toán để đưa ra đáp án.

  + Chi phí về không gian: kích thước bộ nhớ để lưu trữ các dữ liệu sử dụng khi thực hiện thuật toán.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖KHÁI NIỆM

<u>Ví dụ</u>: giả sử một thuật toán có độ phức tạp về thời gian theo kích thước đầu vào n được ước lượng là n^2, điều đó có nghĩa là số lượng phép toán có tốc độ tăng theo hàm số n^2 nếu kích thước đầu vào tăng theo n.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

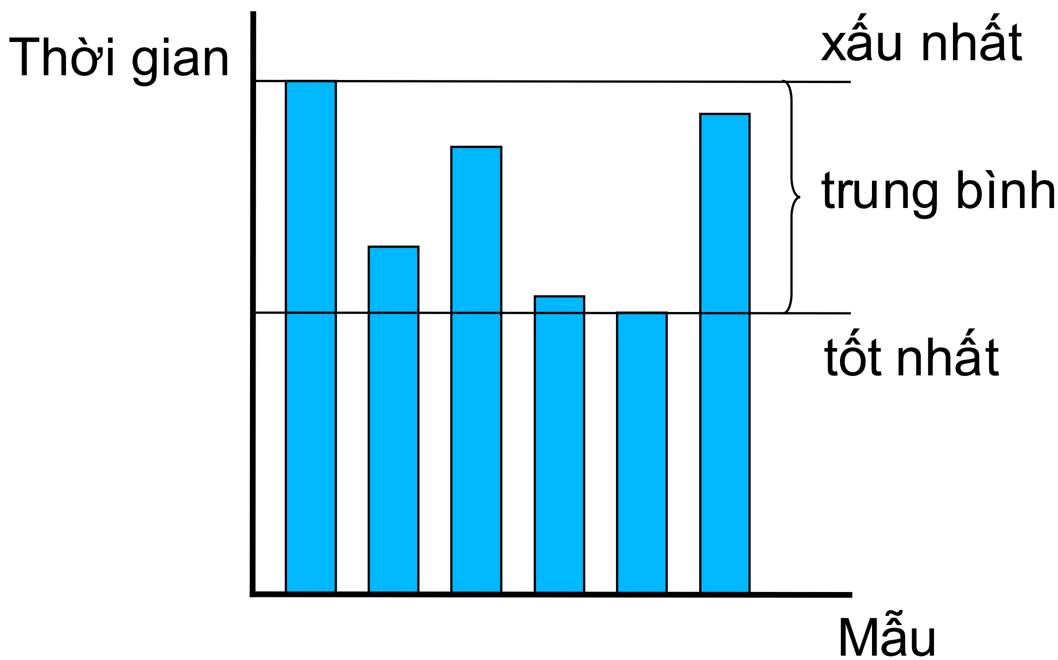
❖KHÁI NIỆM

- Độ phức tạp tính toán được ước lượng để so sánh giữa hai thuật toán cho một bài toán nhằm chọn lựa thuật toán hiệu quả hơn.
- Giữa chi phí về thời gian và chi phí không gian có mối quan hệ tỉ lệ nghịch. Một thuật toán nếu giảm được chi phí về thời gian thì phải tăng chi phí về không gian, và ngược lại.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖KHÁI NIỆM

- Độ phức tạp tính toán có thể được ước lượng theo ba trường hợp: trường hợp xấu nhất, trường hợp trung bình và trường hợp tốt nhất.



ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

- PHÉP TÍNH

  Phép tính là một công việc máy tính thực hiện có thời gian xác định và không phụ thuộc vào kích thước dữ liệu đầu vào của bài toán.

Ví dụ: một số phép tính như tính toán số học, phép gán và các lời gọi hàm xuất nhập cơ bản.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖PHÉP TÍNH

Ví dụ: không phải phép tính:

```
void sapxep(int a[], n) {//...}
void main() {
//...
sapxep(a, n);
//...
}
```

lời gọi hàm sapxep(a, n) không phải là một phép tính.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖TÍNH SỐ LƯỢNG PHÉP TÍNH

Phương pháp tính số lượng phép tính phụ thuộc vào chiến lược đánh giá thuật toán. Có 3 chiến lược đánh giá thuật toán:

- Đánh giá theo trường hợp tốt nhất
- Đánh giá theo trường hợp xấu nhất.
- Đánh giá theo trường hợp trung bình.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖ TÍNH SỐ LƯỢNG PHÉP TÍNH

Thông thường, với mục đích đánh giá chung, một thuật toán sẽ được đánh giá theo trường hợp xấu nhất. Mục đích là để xác định thuật toán có khả thi hay không. Việc đánh giá theo trường hợp xấu nhất có các đặc điểm sau:

- Tính toán đơn giản.
- Không ước lượng được sự hiệu quả.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖TÍNH SỐ LƯỢNG PHÉP TÍNH

Cho n là kích thước dữ liệu đầu vào, số lượng phép tính của một thuật toán được tính theo các quy tắc sau:

- Số phép tính cho một công việc là

$$f_A(n) = 1.$$

- Số phép tính cho một công việc C gồm 2 công việc liên tiếp A và B là:

$$f_C(n) = f_A(n) + f_B(n)$$

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖TÍNH SỐ LƯỢNG PHÉP TÍNH

- Số phép tính cho một công việc C chứa hai công việc được lựa chọn A và B là:

$$f_c(n) = \max(f_B(n), f_A(n));$$

<u>Ví dụ 3</u>:

A = { a = 100; }, $f_A(n) = 1$.

B = { a = a + 12; }, $f_B(n) = 1$.

C = { a = 100; a = a + 12; }, $f_C(n) = f_A(n) + f_B(n) = 2$.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖TÍNH SỐ LƯỢNG PHÉP TÍNH

- Số phép tính cho một công việc đệ quy A bằng tổng số phép tính tại các bước gọi đệ quy.

Ví dụ:

```
int gt(int n) {
    if (n == 1) return 1;
    else return n * gt(n - 1);
}
```

f(1) = 1 + 1 = 2

f(2) = 1 + 2 + f(1)

f(3) = 1 + 2 + f(2)

...

f(n) = 1 + 2 + f(n - 1)

= 3n - 1

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖TÍNH SỐ LƯỢNG PHÉP TÍNH

Ví dụ 4:

```
D =   {   if a = 10
          a = 100;
          else {
                a = 100;
                a = a + 12;
          }
      }
```

$f_D(n) = max(f_A(n),f_C(n)) + 1 = 3.$

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖TÍNH SỐ LƯỢNG PHÉP TÍNH

Ví dụ 5: tính số lượng phép tính của đoạn chương trình A sau:

```
int s = 0;
for (i =1; i <= n; i++) {
    s = s + i;
}
```

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖TÍNH SỐ LƯỢNG PHÉP TÍNH

- 1 phép gán s = 0 → số lượng phép tính: 1
- 1 phép gán i = 1 → số lượng phép tính: 1
- liên tiếp n lần phép tính s = s + i

  → số lượng phép tính: 1+..+1 = n
- liên tiếp n lần phép kiểm tra i <= n:

  → số lượng phép tính: n
- liên tiếp n lần phép tăng i++:

  → số lượng phép tính: n

→ Số lượng phép tính của đoạn chương trình: 3n+2.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖TÍNH SỐ LƯỢNG PHÉP TÍNH

Ví dụ 6: tính số lượng phép tính của đoạn chương trình A sau:

```
for (i =1; i < n; i++)
  for (j=i+1; j<=n; j++)
      if (a[i] > a[j]) {
          int tmp = a[i];
          a[i] = a[j];    a[j] = a[i];
      }
```

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖TÍNH SỐ LƯỢNG PHÉP TÍNH

- Đặt đoạn chương trình C là

```
if (a[i] > a[j]) {
    int tmp = a[i];
    a[i] = a[j];     a[j] = a[i];
}
```

- Đặt đoạn chương trình B là

```
for (j=i+1; j<=n; j++)
C
```

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖TÍNH SỐ LƯỢNG PHÉP TÍNH

- Số lượng phép tính của C là $f_C(n) = 4$.
- Số lượng phép tính của B là

$$f_B(n) = (n-i).(f_C(n) + 1) +1 \ + 2(n-i)= 7(n-i) + 1.$$

- Số lượng phép tính của A là

$$f_A(n) \quad = 7(n-1) + 1 + 7(n-2) + 1 + \dots + 7.1 + 1 + 2(n-1)+1$$

$$= 7[(n-1) + (n-2) + \dots + 1] + 3(n-1)$$

$$= 7n(n-1)/2 + 3(n-1)$$

$$= (7n^2 - 7n + 6n - 6) / 2$$

$$= (7n^2 - n - 6) / 2$$

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

Cho một thuật toán A thực hiện trên dữ liệu đầu vào có kích thước n. Chi phí về thời gian để thực hiện giải thuật này là f(n). Khi đó, độ phức tạp tính toán của thuật toán có thể được ước lượng theo các bậc như sau:

* **Bậc Big-O (Ô lớn)**

- Độ phức tạp của thuật toán A là O(g(n)) nếu tồn tại các hằng số c>0, n_0>0 sao cho

$$f(n) \le c.g(n) \; \forall \; n \ge n_0$$

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

Ví dụ 7: với đoạn chương trình A trong Ví dụ 6, ta có:

$$f_A(n) = (7n^2 - n - 6) / 2$$

Xét hàm số $g(n) = n^2$

Với $c = 4$, $n_0=1$, ta có:

$$4.g(n) = 4.n^2 \geq f_A(n), \forall n \geq n_0$$

Vậy, thuật toán A có độ phức tạp tính toán là $O(n^2)$

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

- Bậc Big-O thường được sử dụng trong đánh giá độ phức tạp tính toán của thuật toán để ước lượng độ phức tạp không thể cao hơn của thuật toán. Hàm g(n) được gọi là giới hạn trên của f(n).
- Lưu ý: nếu độ phức tạp tính toán của thuật toán A là O(g(n)), thì độ phức tạp của thuật toán A cũng là O(h(n)) với h(n) là giới hạn trên của g(n)

Ví dụ 8: giả sử thuật toán A có số phép tính là

$f(n)=n^2+2n+1$, khi đó, độ phức tạp tính toán của A có thể là $O(n^2)$, $O(n^3)$, $O(2^n)$, $O(n^n)$,...

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

* **<u>Bậc Big-Ω (Ômêga lớn)</u>**

- Độ phức tạp của thuật toán A là Ω(g(n)) nếu tồn tại các hằng số c>0, n_0>0 sao cho

$$f(n) \geq c.g(n) \ \forall \ n \geq n_0$$

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

Ví dụ 9: với đoạn chương trình A trong Ví dụ 6, ta có:

$$f_A(n) = (7n^2 - n - 6) / 2$$

Xét hàm số $g(n) = n^2$

Với $c=1$, $n_0=2$, ta có:

$$g(n) = n^2 \leq f_A(n), \forall n \geq n_0$$

Vậy, thuật toán A có độ phức tạp tính toán là $\Omega(n^2)$

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖ CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

- Bậc Big-Ω dùng để ước lượng độ phức tạp không thể thấp hơn của thuật toán. Hàm g(n) được gọi là giới hạn dưới của f(n).

- Lưu ý: nếu độ phức tạp tính toán của thuật toán A là $\Omega(g(n))$, thì độ phức tạp của thuật toán A cũng là $\Omega(h(n))$ với h(n) là giới hạn dưới của g(n).

Ví dụ 10: giả sử thuật toán A có số phép tính là

$f(n)=n^n + n^2 + 1$, khi đó, độ phức tạp tính toán của A có thể là $\Omega(n^n)$, $\Omega(2^n)$, $\Omega(n^3)$, $\Omega(n^2)$.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

* **Bậc Big-Θ (Têta lớn)**

- Độ phức tạp của thuật toán A là $\Theta(g(n))$ nếu tồn tại các hằng số $c_0>0$, $c_1>0$, $n_0>0$ sao cho

$$c_0.g(n) \le f(n) \le c_1.g(n) \ \forall\ n \ge n_0$$

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

Ví dụ 11: với đoạn chương trình A trong Ví dụ 6, ta có:

$$f_A(n) = (7n^2 - n - 6) / 2$$

Xét hàm số $g(n) = n^2$

Với $c_0 = 1$, $c_1 = 4$, $n_0=2$, ta có:

$$g(n) = n^2 \leq f_A(n) \leq 4n^2 = 2.g(n), \forall n \geq n_0$$

Vậy, thuật toán A có độ phức tạp tính toán là $\Theta(n^2)$

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

- Bậc Big-Θ dùng để ước lượng độ phức tạp tương đương của thuật toán. Hàm g(n) được gọi là giới hạn chặt của f(n).

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

* **<u>Một số tính chất</u>**

- Nếu thuật toán A có số phép tính dựa trên kích thước đầu vào n là một đa thức P(n) bậc k, khi đó, độ phức tạp tính toán của A là có thể là $O(n^k)$, $\Omega(n^k)$, $\Theta(n^k)$.
- Nếu thuật toán A có số phép tính dựa trên kích thước đầu vào n là $\log_a f(n)$,

  do $\log_a f(n)=\log_a b.\log_b f(n)$

  nên độ phức tạp tính toán của A có thể ghi là O(log f(n)) mà không cần ghi cơ số. Điều này cũng đúng với bậc Ω và Θ.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

* **Một số tính chất**

- Nếu thuật toán A có hai công việc liên tiếp nhau T1 và T2 lần lượt có độ phức tạp là O(f(n)) và O(g(n)) thì độ phức tạp của A: O(max(f(n),g(n))).
- Nếu thuật toán A có hai công việc T1 lồng T2, T1 và T2 lần lượt có độ phức tạp là O(f(n)) và O(g(n)) thì độ phức tạp của A: O(f(n).g(n)).

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖CÁC BẬC ĐÁNH GIÁ THUẬT TOÁN

* **<u>Nhận xét:</u>**

- Trong các ký pháp để đánh giá thuật toán, Bậc Big-Θ thể hiện độ phức tạp tính toán tốt nhất
- Trong những trường hợp không thể xác định được giới hạn chặt của hàm số biểu diễn số lượng phép tính của thuật toán, người ta thường dùng bậc Big-O để thể hiện độ phức tạp tính toán.
- Khi đánh giá thuật toán, người ta sử dụng hàm giới hạn g(n) đơn giản và sát với f(n) nhất có thể

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖SỰ PHÂN LỚP CÁC THUẬT TOÁN

Độ phức tạp	Tên gọi độ phức tạp tương ứng
O(1)	Độ phức tạp hằng số
O(log(n))	Độ phức tạp logarit
O(n)	Độ phức tạp tuyến tính
O(nlog(n))	Độ phức tạp nlogn
$O(n^k)$	Độ phức tạp đa thức
$O(k^n)$	Độ phức tạp hàm mũ
O(n!)	Độ phức tạp giai thừa

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖ PHÂN TÍCH TRƯỜNG HỢP TRUNG BÌNH

Phân tích trường hợp trung bình được thực hiện theo trình tự sau:

- Tính số phép toán được thực hiện trung bình.
- Dùng các bậc đánh giá để ước lượng.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

- PHÂN TÍCH TRƯỜNG HỢP TRUNG BÌNH

  Số phép toán trung bình được tính dựa vào giả thiết các trường hợp đều có xác suất như nhau và số phép toán trung bình chính là số phép toán kỳ vọng của các trường hợp.

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

- PHÂN TÍCH TRƯỜNG HỢP TRUNG BÌNH

Ví dụ: đánh giá độ phức tạp trung bình của thuật toán sau:

```
for (i = 0; i < n; i++) {
    if (a[i] == x) return i;
}
```

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

- PHÂN TÍCH TRƯỜNG HỢP TRUNG BÌNH

Số phép tính của thuật toán trên trong trường hợp giá trị x nằm ở vị trí thứ i, i=0,..,n là:

$T_i = 1 + (i+1) + i + i + 1 = 3i + 3 = 3(i + 1)$

Với giả thiết các trường hợp này là ngẫu nhiên, nên số phép tính trung bình của thuật toán:

$T = \Sigma_i\ 3(i + 1)/n,\ i = 0..n$

$= (3/n)\ \Sigma_i\ (i + 1) = 3(n(n+1) + 2n)/(2n)$

$= 3(n + 3) / 2$

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

- PHÂN TÍCH TRƯỜNG HỢP TRUNG BÌNH

Vậy độ phức tạp trung bình của thuật toán là O(n)

ĐỘ PHỨC TẠP CỦA GIẢI THUẬT

❖MỘT SỐ CÔNG THỨC

$1 + 2 + 3 + .. + n = n(n + 1)/2$

$1 + 3 + 5 + .. + (2n - 1) = n^2$

$a^0 + a^1 + a^2 + .. + a^n = (a^{n+1} - 1)/(a - 1)$

$1.2 + 2.3 + 3.4 + .. + n(n + 1) = n(n+1)(n+2) / 3$

$1^2 + 2^2 + 3^2 + .. + n^2 = n(n + 1)(2n + 1) / 6$

$1^3 + 2^3 + 3^3 + .. + n^3 = n^2(n + 1)^2 / 4$

$1^k + 2^k + 3^k + .. + n^k \approx n^{k+1} / (k + 1), k \neq -1$