Phân tích và Thiết kế THUẬT TOÁN

Nguyễn Mậu Uyên

uyennm@mta.edu.vn

Web: fit.mta.edu.vn/~uyennm

Bài 2 - Đánh giá độ phức tạp thuật toán

PHÂN TÍCH VÀ THIẾT KẾ THUẬ TOÁN

NỘI DUNG

I. Giới thiệu
II. Phân tích trực tiếp các đoạn mã
III. Phân tích đoạn mã có lời gọi chương trình con
IV. Đánh giá dựa trên thực nghiệm
V. Bài tập

1. Giới thiệu

- Trước khi thực hiện tính độ phức tạp thuật toán A giải bài toán P ta cần – f(n):
  - Xác định độ dài dữ liệu - n: có thể là số ký tự, số phần tử của mảng, ….
  - Tiêu chí đánh giá: thống nhất là số các thao tác cơ bản (gán, so sánh..)
- Để đánh giá có thể sử dụng:
  - Phân tích trực tiếp để tính số các thao tác
  - Phương pháp đệ quy

1. Giới thiệu

- Dựa trên một số quy tắc
  - Quy tắc cộng
  - Quy tắc nhân
  - Quy tắc phân tích một số câu lệnh
  - Xét tính chất của chương trình con

1. Giới thiệu

- Quy tắc cộng
  - T1(n) và T2(n) là thời gian thực hiện của hai đoạn chương trình con nối tiếp nhau (độc lập) P1, P2 và
  - T1(n)= O(f1(n)); T2(n)=O(f2(n))
  - Khi đó thời gian (độ phức tạp thời gian) thực hiện của 2 đoạn chương trình đó là

$$T(n)=T1(n)+T2(n) = O(\max\{f1(n), f2(n)\}$$

Chứng minh: Theo đầu bài, tồn tại các hằng M1, M2, n1, n2 để

$$T1(n) \le M1*f1(n),\ \forall n>n1,\ \ T2(n) \le M2*f2(n),\ \forall n>n2$$

Khi đó

$$T(n) = T1(n) + T2(n) \le M1*f1(n)+M2*f2(n),$$

$$\le M.f(n) \text{ với } n>n0,\ \ M=\max(M1,M2),\ n0=\max(n1,n2)$$

$$f(n)=\max(f1(n),f2(n))$$

1. Giới thiệu

- Quy tắc nhân
  - T1(n) và T2(n) là thời gian thực hiện của hai đoạn chương trình con lồng nhau (phụ thuộc) P1, P2 và
  - T1(n)= O(f1(n)); T2(n)=O(f2(n))
  - Khi đó thời gian (độ phức tạp thời gian) thực hiện của 2 đoạn chương trình đó là

$$T(n)=T1(n)*T2(n) = O(f1(n)*f2(n))$$

Chứng minh: (tương tự với quy tắc cộng)

1. Giới thiệu

- Quy tắc phân tích câu lệnh
  - Các câu lệnh đơn (gán, đọc, ghi…) có độ phức tạp là Hằng - O(1)
  - Ví dụ:

    (1) - read(a)

    (2) - read(b)

    (3) - read(c)

    (4) - delta = b*b – 4*a*c
  - Nhận xét: Trong đoạn chương trình chỉ bao gồm các lệnh đơn kế tiếp nhau (không chứa các vòng lặp), theo quy tắc cộng => Độ phức tạp thuật toán là hằng O(1)

1. Giới thiệu

- Quy tắc phân tích câu lệnh
  - Cấu trúc if: thời gian kiểm tra điều kiện + thời gian thực hiện sau THEN hoặc ELSE
  - Cấu trúc lặp:
    - thời gian thực hiện vòng lặp là tổng thời gian thực hiện của thân vòng lặp.
    - Nếu số bước tính trong vòng lặp không đổi (theo mỗi bước lặp) thì thời gian thực hiện vòng lặp bằng tích của số lần lặp nhân với thời gian thực hiện thân vòng lặp.

2. Phân tích trực tiếp

1. Phương pháp chung:

   Phân tích trực tiếp đoạn mã và sử dụng các kỹ thuật:

   - Phép đếm
   - Tính tổng hữu hạn
   - Xét dấu hàm
   - ...

   Xác định số phép toán chủ yếu

   Phép toán chủ yếu trong các đoạn mã là phép gán và so sánh.

   Phương pháp này không giải quyết được tất cả các trường hợp.

2. Phân tích trực tiếp

- Ví dụ 1:

```
s = 0
i = 1
while  i≤n do
     j = n - i
     while  j≥1 do
               s = s + 1
               j = j - 1
     endw
     i = i + 1
endw
```

Khảo sát độ phức tạp trên số phép gán và so sánh trong thuật toán.

2. Phân tích trực tiếp

```
s = 0                              } ? phép gán
i = 1
while i≤n do                       → ? phép so sánh
    j = n - i                      → ? phép gán
    while j≥1 do                   → ? phép so sánh
            s = s + 1              → ? phép gán      P(i)
            j = j - 1              → ? phép gán
    endw
    i = i + 1                      → ? phép gán
endw
```

$$\text{Số phép gán} = 2 + n + \left[\sum_{i=1}^{n} \text{Gán}(P_i)\right] + n$$

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

2. Phân tích trực tiếp

Số phép so sánh = ? (Bài tập 1)

- Ví dụ 2:

```
sum = 0
i = 1
while i≤n do
        j = n-i*i
        while j ≤ i*i do
                sum=sum + i*j
                j=j+1
        endw
        i=i+1
endw
```

P_i

$$\text{Số phép gán} = 2 + n + \left[\sum_{i=1}^{n} \text{Gán}(P_i)\right] + n$$

2. Phân tích trực tiếp

$$= 2 + 2n + \sum_{i=1}^{n} (2\alpha_i) = 2 + 2n + 2\sum_{i=1}^{n} \alpha_i$$

Nếu thay dòng lệnh j=n-i*i bằng dòng lệnh j=1 thì $\alpha_i = i^2$

Vòng lặp P_i chỉ thực hiện khi $n - i^2 \le i^2 \Leftrightarrow i^2 \ge n/2$

Từ đây suy ra:

$$\alpha_i = \begin{cases} 0 & \text{nếu } i^2 < \dfrac{n}{2} \\ i^2 - (n - i^2) + 1 & \text{nếu } i^2 \ge \dfrac{n}{2} \end{cases}$$

Bài tập 2: Hãy viết chương trình thử nghiệm để đếm số phép gán và so sánh của đoạn chương trình ví dụ 2, để kiểm tra lại lý thuyết.

Như vậy:

$$\sum_{i=1}^{n} \alpha_i = \sum_{i=\left\lceil \sqrt{\frac{n}{2}} \right\rceil}^{n} (2i^2 - n + 1) = 2\sum_{i=\left\lceil \sqrt{\frac{n}{2}} \right\rceil}^{n} i^2 - \left(n - \left\lceil \sqrt{\frac{n}{2}} \right\rceil + 1\right)(n-1)$$

2. Phân tích trực tiếp

- Ví dụ 3: Xét thuật toán tìm phần tử max của mảng một chiều có n phần tử.

```
max = A[0];
i=1;
while i<n do
          if  max< A[i] then
                    max = A[i];
          endif
          i=i+1;
endw
```

Số phép so sánh = ?
Số phép gán = ?

$$\alpha(n) = \begin{cases} 0 & \text{tối thiểu khi A[0] là max} \\ n-1 & \text{tối đa khi A được sắp xếp tăng hay max nằm ở cuối} \\ ? & \text{Trung bình: dùng công cụ toán} \end{cases}$$

ss = n + n – 1 = 2n - 1

gn =n + 1 + α(n) = 2n (xấu nhất)

2. Phân tích trực tiếp

1. Những tính toán lặp
   – Tùy tình huống
2. Các loại tính toán lặp
   – Số lần lặp xác định tường minh: được thể hiện rõ ràng trong đoạn mã. Có thể tính toán bằng một công thức xác định.

   Ví dụ: Tổng n số nguyên.

   – Số lần lặp không tường minh: biến sẽ ngẫu nhiên phụ thuộc vào dữ liệu đầu vào và phân bố.

   Ví dụ: Tìm số lớn nhất.

2. Phân tích trực tiếp

- Ví dụ 4: Xét đoạn mã.

```
i=1;
res=0;
while i≤n do
        j=1;
        k=1;
        while j ≤ i do
                res=res+i*j;
                k=k+2;
                j=j+k;
        endw
        i=i+1;
endw
```

P_i

2. Phân tích trực tiếp

- Vòng lặp while ngòai cùng: số lần lặp tường minh: n lần .
- Vòng lặp while bên trong: số lần lặp không xác định. Cách giải quyết:
  - Gọi α_i là số lần lặp của vòng while này (quy ước tính độc lập).
  - Việc xác định số phép gán, so sánh trong đoạn mã sẽ quy về tính $\sum_{i=1}^{n} \alpha_i$ theo α_i
- Ta thấy j là tổng với các số là 1, 3, 5, ... Nên là các số chính phương.
  - $\Rightarrow \alpha_i$ là số phần tử của $\{r/\ r \geq 1\ \&\ r^2 \leq i\}$.
  - $\Rightarrow \alpha_i = \lfloor \sqrt{i} \rfloor$

2. Phân tích trực tiếp

- Ví dụ 5: Xét đoạn mã tương tự ví dụ 4.

```
i=1;
res=0;
s=0; // số thực
while i≤n do
        j=1;
        s=s+1/i;
        while j ≤ s do
                res=res+i*j;
                j=j+1;
        endw
        i=i+1;
endw
```

P_i

Lúc này $\alpha_i = \lfloor H_i \rfloor$

với $H_i = 1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{i}$, H_i là số điều hòa

2. Phân tích trực tiếp

- Ví dụ 6: xét đoạn mã

Đoạn chương trình dừng khi nào?

```
i=0;
A[n]=x;
while  A[i] ≠ x do
            i=i+1;
endw
```

Đoạn chương trình dừng trong các trường hợp sau:

- $i=n \Leftrightarrow x \neq A[i], \forall i \in \{0,1, \ldots, n-1\}$
- $i<n \Leftrightarrow \exists\, i_0 \in \{0,1, \ldots, n-1\}$ sao cho $x=A[i_0]$ và $x \neq A[j], \forall j<i_0$

Vậy số lần lặp không xác định tường minh, nhưng lại tường minh cho một mảng dữ liệu cụ thể

2. Phân tích trực tiếp

- Rẽ nhánh tất định:
  - Cân bằng cách nhánh
  - Độ lệch các nhánh rẽ tính được
  - Không phụ thuộc dữ liệu nhập
- Rẽ nhánh phụ thuộc phân bố dữ liệu:
  - Phải tính toán theo xác suất phân bố của dữ liệu

2. Phân tích trực tiếp

- Ví dụ 7: Tìm số lớn nhất trong mảng một chiều.

```
i=1;
max=A[0];
A[n]=x;
while  i<n do
        if max<A[i] then
                max=A[i];
        endif
        i=i+1;
endw
```

Biến α_n là biến ngẫu nhiên lấy các giá trị
Rời rạc {0, 1, 2, ..., n-1}

2. Phân tích trực tiếp

- Ví dụ 8:

```
s=0;
i=1;
while  i≤n do
        j=1;
        while j≤i² do
                s=s+i*j;
                j=j+1;
        endw
        i=i+1;
endw
```

P_i

$$\text{số phép so sánh} = n+1+\sum_{i=1}^{n} P(i)\Big|_{\text{so sánh}} = n+1+\sum_{i=1}^{n}(i^2+1)$$

$$= 2n+1+\frac{n(n+1)(2n+1)}{6} \quad \sim n^3$$

$$\text{số phép gán} = 2n+2+\sum_{i=1}^{n} P(i)\Big|_{\text{so sánh}} = ?$$

2. Phân tích trực tiếp

- Ví dụ 9:

```
count=0;
i=n;
while  i>0 do
            count=count +  i%2;
            i=i/2;
endw
```

So sánh = α +1
Gán = 2 +2α

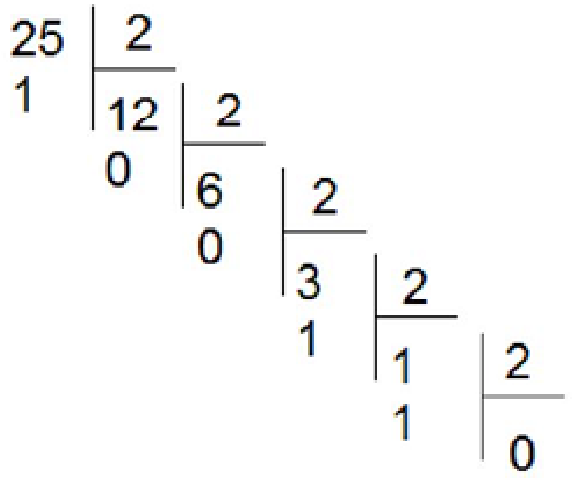
→ Cần ước lượng α

$$\text{Có } \alpha(\text{n}) \approx \log_2 n$$

2. Phân tích trực tiếp

Ví dụ: có n=25 → số lần lặp?

= số chữ số trong biểu diễn nhị phân của n.



$n=25=1x2^4 + 1x2^3 + 0x2^2 + 0x2^1 + 1x2^0$

$n=b_k\, b_{k-1} \ldots b_1\, b_0$ với $b_i = \{0,1\}$

$\alpha = k+1$

2. Phân tích trực tiếp

- Ví dụ 10:

```
sum=0;
i=1;
while  i≤n do
          j=i;
          while j>0 do
                    sum=sum + 1;
                    j=j/2;
          endw
          i=i + 1;
endw
```

P_i lặp n lần

So sánh = $n+1 + \sum_{i=1}^{n} P_i = n+1+\sum_{i=1}^{n}(\alpha(i)+1) = 2n+1+\sum_{i=1}^{n}\alpha(i)$

Xác định $\alpha(i)$? Bài tập 3

Gán = ? Bài tập 4

2. Phân tích trực tiếp

- Ví dụ 11:

```
max=A[0];
i=1;
count=0;
while  i≤n do
        if (max<A[i])
                max =A[i];
        else
                count=count +1;
        endif
        i = i+1;
endw
```

P_l lặp?

Gán = ?
So sánh =?

Bài tập 5

2. Phân tích trực tiếp

- Ví dụ 12:

So sánh = ?
Gán = ?

```
i=1;
c_d =0;
c_a =0;
c_z =0;
while  i≤n do
        if (A[i]>0)
                c_d =c_d+1;
        else
                if(A[i]<0)
                        c_a=c_a+1;
                else
                        c_z=c_z +1;
                endif
        endif
        i = i+1;
endw
```

P_i lặp?

2. Phân tích trực tiếp

- Ví dụ 12:

So sánh = ?
Gán = ?

```
i=1;
c_d =0;
c_a =0;
c_z =0;
while  i≤n do
        if (A[i]>0)
                c_d =c_d+1;
        else
                if(A[i]<0)
                        c_a=c_a+1;
                else
                        c_z=c_z +1;
                endif
        endif
        i = i+1;
endw
```

P_i lặp?

Nếu viết lại P_i ta có:

```
if(A[i]>0)
        c_d=c_d+1;
else
        if(A[i]<0)
                c_a=c_a+1;
        endif
        if(A[i]==0)
                c_z=c_z+1;
        endif
endif
```

Gán =?
So sánh =?

2. Phân tích trực tiếp

- Ví dụ 12:

So sánh = ?
Gán = ?

```
i=1;
c_d =0;
c_a =0;
c_z =0;
while i≤n do
        if (A[i]>0)
                c_d =c_d+1;
        else
                if(A[i]<0)
                        c_a=c_a+1;
                else
                        c_z=c_z +1;
                endif
        endif
        i = i+1;
endw
```

P_i lặp?

Nếu viết lại P_i ta có:

```
if(A[i]>0)
        c_d=c_d+1;
endif
if(A[i]<0)
        c_a=c_a+1;
endif
if(A[i]==0)
        c_z=c_z+1;
endif
```

Gán =?
So sánh =?

2. Phân tích trực tiếp

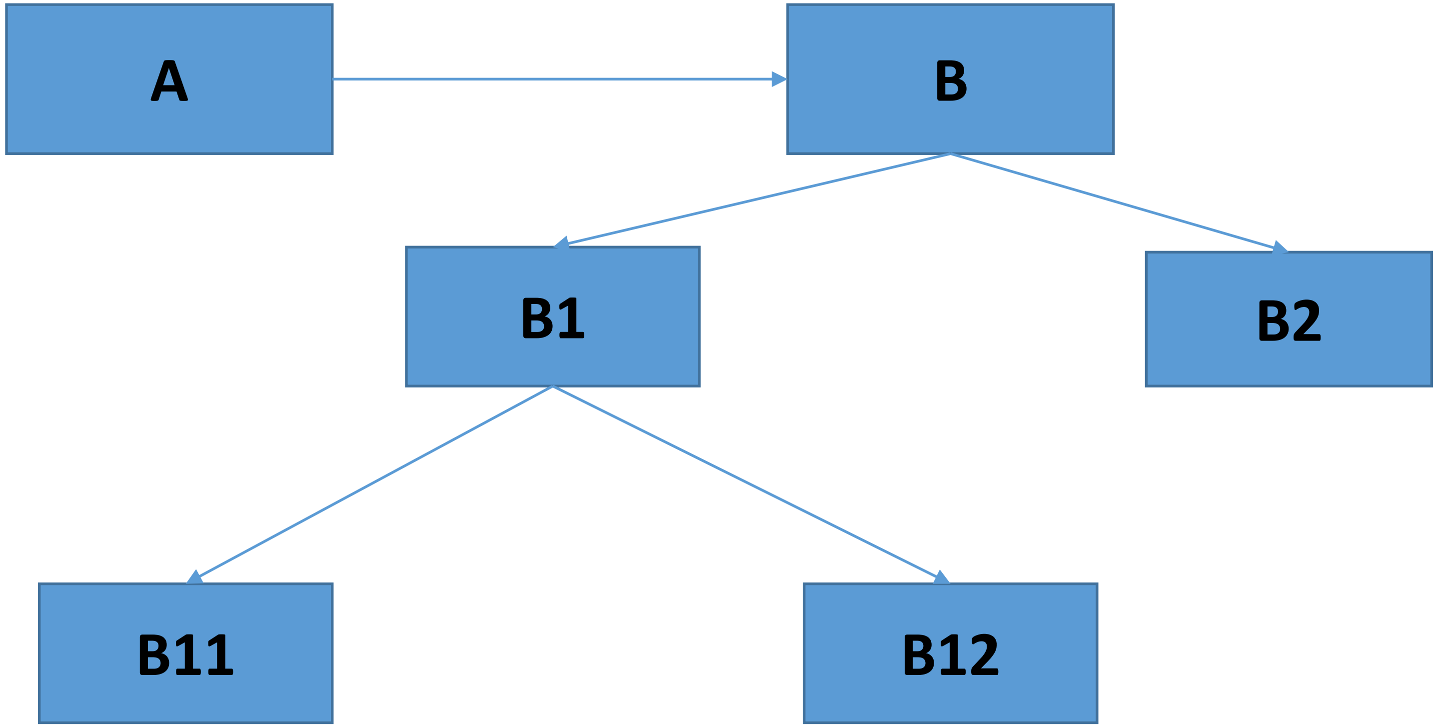
- Ví dụ 13:

```
found = false;
i =1;
sum=0;
while i≤n do
        if((!found)&&(A[i]==X))
                idx_f=i;
                found=true;
        endif
        sum=sum+A[i];
        i=i+1;
endw
```

- Khi $x \in \{A[i] / i=1..n\}$
  - Gán = 5 + 2n
  - So sánh =3n +1
- Khi $x \notin \{A[i] / i=1..n\}$
  - Gán = 3 + 2n
  - So sánh =3n +1

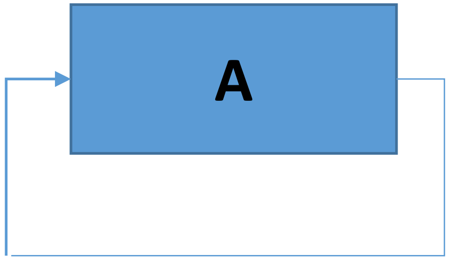
3. Đoạn chương trình có gọi chương trình con

- Gọi chương trình con không đệ quy



3. Đoạn chương trình có gọi chương trình con

- Gọi chương trình con đệ quy



Tính thời gian thực hiện của A?

3. Đoạn chương trình có gọi chương trình con

- Độ phức tạp chương trình con dạng đệ quy

  - Cách giải quyết:
    1. Thành lập phương trình đệ quy
    2. Giải phương trình đệ quy

    Nghiệm của lời giải ở bước 2 là thời gian thực hiện chương trình

3. Đoạn chương trình có gọi chương trình con

- Độ phức tạp chương trình con dạng đệ quy

  - Phương trình đệ quy: Biểu diễn mỗi liên hệ giữa T(n) với T(k), k<n. Trong đó T(n) thời gian thực hiện chương trình và T(k) thời gian thực hiện với kích thước bộ dữ liệu là k, và k<n.
  - Để lập phương trình: Căn cứ vào chương trình đệ quy.

3. Đoạn chương trình có gọi chương trình con

- Độ phức tạp chương trình con dạng đệ quy

  - Dạng tổng quát:

$$T(n) = \begin{cases} C(n_0), & \text{với } n=n_0 \\ T(k) + d^* & \text{với } n>k>n_0 \end{cases}$$

    - $C(n_0)$: Thời gian thực hiện khi $n=n_0$
    - $T(k)$: thời gian thực hiện khi $n>k>n_0$
    - d^*: Thời gian phân chia và tổng hợp kết quả

3. Đoạn chương trình có gọi chương trình con

- Độ phức tạp chương trình con dạng đệ quy
  - Ví dụ: xét hàm tính giai thừa

```
Function gt(n)
begin
    if n=0 then gt=1
    else gt=n*gt(n-1)
end
```

Gọi T(n) là thời gian tính n!, thì T(n-1) là thời gian tính (n-1)!

Khi n=0, ta có C(0)=1 (phép gán)

3. Đoạn chương trình có gọi chương trình con

- Độ phức tạp chương trình con dạng đệ quy
  - Ví dụ: xét hàm tính giai thừa

```
Function gt(n)
begin
    if n=0 then gt=1
    else gt=n*gt(n-1)
end
```

Khi n>0, hàm gọi đệ quy gt(n-1), tốn T(n-1)

Tổng hợp kết quả ở đây cần 1 phép gán, d*=1

3. Đoạn chương trình có gọi chương trình con

- Độ phức tạp chương trình con dạng đệ quy
  - Ví dụ: xét hàm tính giai thừa

```
Function gt(n)
begin
    if n=0 then gt=1
    else gt=n*gt(n-1)
end
```

$$T(n) = \begin{cases} 1, & \text{với } n=0 \\ T(n-1) + 1 & \text{với } n>k>0 \end{cases}$$

Khi n>0, hàm gọi đệ quy gt(n-1), tốn T(n-1)

Tổng hợp kết quả ở đây cần 1 phép gán, d*=1

3. Đoạn chương trình có gọi chương trình con

- Độ phức tạp chương trình con dạng đệ quy
  - Giải phương trình đệ quy – Phương pháp truy hồi

1. Với $n>k>n_0$: dùng phương trình đệ quy lần lượt thay thế T(k) vào vế phải
2. Dừng khi $k=n_0$
3. Thế $T(n_0)$ để tìm T(n)

3. Đoạn chương trình có gọi chương trình con

- Độ phức tạp chương trình con dạng đệ quy
  - Giải phương trình đệ quy – Phương pháp truy hồi

1. Ví dụ: Giải

$$T(n) = \begin{cases} 1, & \text{với } n=0 \\ T(n-1) + 1 & \text{với } n>k>0 \end{cases}$$

T(n) = T(n-1) + 1

= T(n-2) + 1 + 1

....

= T(n-i) + i

Dừng khi n-i = 0, hay i=n, khi đó T(n) = 1 + n = O(n)

3. Đoạn chương trình có gọi chương trình con

- Độ phức tạp chương trình con dạng đệ quy
  - Giải phương trình đệ quy – Phương pháp truy hồi

1. Ví dụ: Giải

$$T(n) = \begin{cases} 0 & \text{với } n=1 \\ T(n/2) + 1 & \text{với } n>1 \end{cases}$$

$T(n) = T(n/2) + 1$

$= T(n/2^2) + 1 + 1$

....

$= T(n/2^i) + i$

Dừng: $n/2^i = 1$ (n_0), hay $i=\log_2 n$, khi đó $T(n) = 0 + \log_2 n$

4. Đánh giá bằng thực nghiệm

- Chèn thêm lệnh đếm trong đoạn mã
- Phát sinh dữ liệu để thực thi đoạn mã
- Ghi xuống file (dạng văn bản)
- Dùng Excel vẽ đồ thị tính phương sai, độ lệch chuẩn → ước lượng độ phức tạp.

4. Đánh giá bằng thực nghiệm

Ví dụ: Thuật toán tìm giá trị lớn nhất

```
max = A[0];
i=1;
while i<n do
    if(max<A[i])
        max=A[i];
    endif
endw
```

1. Cài đặt hàm

```
int findMax(int n, int a[]){
    ...
}
```

4. Đánh giá bằng thực nghiệm

2. Cài đặt đếm

```
int evaluateFindMax(int n, int a[], long &gan, long &sosanh){
    int max=a[0];
    int i=1;
    gan=2;
    sosanh=0;
    while(i<n){
            sosanh+=2;
            if(max<a[i]){
                    max=a[i];
                    gan++;
            }
            i++;
            gan++;
    }
    sosanh++;
    return max;
}
```

4. Đánh giá bằng thực nghiệm

3. Phát sinh dữ liệu

```
void  generateData(int n, int *a){
      // dùng hàm random hay rank hoặc kết hợp nhiều // hàm, hay tự
      viết (sách)
}
```

4. Chạy thử nghiệm và ghi dữ liệu

```
#define ẩ MAX 50
#define ẩ LOOP 200
int a[ẩ MAX];
void runData(char *name){
      FILE *fp  = fopen(name,"wt");
      if(fp==ẩ ULL){
                printf("Can not open to write file!!!");
                return;
      }
```

4. Đánh giá bằng thực nghiệm

```
    int n=1;
    while(n<ả LOOP){
            long gan=0;
            long sosanh=0;
            generateData(ả MAX,a);
            evaluteFindMax(ả MAX,a,gan,sosanh);
            fprintf(fp,"%d\t%e\t%e\n",n,gan,sosanh);
            n++;
    }
    fclose(fp);
}
```

4. Đánh giá bằng thực nghiệm

Chú ý

- Phân biệt rõ ràng: phép gán, so sánh khóa, sao chép mẫu tin, so sánh
  - Ví dụ khi so sánh khóa là chuỗi k ký tự thì?
  - Sao chép một record sinh viên?
  - Phép hoán đổi 2 phần tử swap(a[i],a[j]):
    - Chỉ là 2 số nguyên → 3 phép gán
    - 2 phần tử bất kỳ?

NỘI DUNG BÀI HỌC

I. Giới thiệu
II. Phân tích trực tiếp các đoạn mã
III. Phân tích đoạn mã có lời gọi chương trình con
IV. Đánh giá dựa trên thực nghiệm
V. Bài tập

5. Bài tập

1. Tính số phép so sánh trong đoạn mã ở ví dụ 1 slide 11.
2. Sử dụng công thức tính tổng dãy lũy thừa tính ra độ phức tạp lý thuyết ở ví dụ 2 slide 13, đánh giá bằng thực nghiệm chương trình trong ví dụ 2 slide 13 và so sánh với đánh giá lý thuyết.
3. Tính tham số α(i) qua đó tính số phép so sánh ở ví dụ 10 slide 26
4. Tính số phép gán ở ví dụ 10 trang 26.
5. Tính số phép so sánh, số phép gán trong đoạn chương trình ở ví dụ 11 slide 27.
6. Tính số phép so sánh, số phép gán trong đoạn chương trình ở ví dụ 12 slide 28.