ĐÁNH GIÁ ĐỘ PHỨC TẠP CỦA THUẬT TOÁN

# Các quy tắc tính độ phức tạp của thuật toán

# Quy tắc cộng

T1 (n) vàT2 (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)}

# Quy tắc nhân

T1(n) vàT2(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))

# Quy tắc phân tích một số câu lệnh

Các câu lệnh đơn (gán, đọc, ghi…) có độ phứctạp là Hằng - O(1)

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)

# Xét tính chất của chương trình con

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.

# Các phương pháp tính

# Phân tích trực tiếp

Phân tích trực tiếp các đoạn mã và sử dụng kỹ thuật để xác định số phép toán chủ yếu (phép gán và so sánh)

**VD1:**

s = 0; 🡪 phép gán

i = 1; 🡪 phép gán

while i <= n do 🡪 phép so sánh

j = n -i; 🡪 phép gán

i = i + 1; 🡪 phép gán

end

**Số phép gán: 2 + n + + n**

**= 2 + n + 2. +n**

**= n2 + n + 2 = O(n2)**

**Giải thích: Trong P(i) có 2 phép gán. Vd với phép gán s = s + 1**

**Vì nếu i = 1, j >= 1 thì phép gán này chạy n-1 lần, nếu i = 2; j>= 1 thì phép gán này chạy n – 2 lần**

**Số phép so sánh: n + n-1 = 2n -1 = O(n)**

# Đánh giá bằng thực nghiệm