Phân tích và thiết kế giải thuật

+ Chương 1: Câu 3, 4, 7, 8, 11, 12

+ Chương 2:

Câu 2. Viết giải thuật Quicksort với pivot là phần tử phải cùng (rightmost)

Câu 13. Closest-pair

+ Chương 3:

Câu 1. Viết giải thuật brute-force, divide-conquer, (decrease-conquer) để tìm vị trí của phần tử nhỏ nhất trong mảng có n phần tử. Phân tích độ phức tạp trong trường hợp xấu nhất và trung bình.

Câu 5. Hiệu chỉnh DFS để phát hiện chu trình. Thảo luận độ phức tạp.

Chương 1

3. Given the following procedure that finds the maximum and minimum elements in an array.

procedure MAXMIN(A, n, max, min)

/\* Set max to the maximum and min to the minimum of A(1:n) \*/

begin

integer i, n;

max := A[1];

min:= A[1];

for i:= 2 to n do

if A[i] > max then max := A[i];

else if A[i] < min then min := A[i];

end

Let C(n) be the complexity function of the above algorithm, which measures the number of element comparisons.

(a) Describe and find C(n) for the worst-case.

(b) Describe and find C(n) for the best-case.

(c) Find C(n) for the average-case when n=3.

a, C(n) là độ phức tạp giải thuật được tính theo thao tác so sánh trong trường hợp xấu nhất.

* Tổng thao tác so sánh của thủ tục MAXMIN chính là số thân vòng lặp được thực thi: 2 \* (n - 1).
* Độ phức tạp giải thuật là O(n).

b, Trường hợp tốt nhất:

* Mảng có 1 phần tử nên MIN = MAX.
* Độ phức tạp giải thuật là O(1).

c,

4. Suppose Module A requires M units of time to be executed, where M is a constant. Find the complexity C(n) of the given algorithm, where n is the size of the input data and b is a positive integer greater than 1.

j:= 1;

while j <= n do

begin

call A;

j := j\*b;

end

- Thông số thiết lập hàm : n

- Tác vụ phân tích : so sánh

- Trường hợp phân tích: xấu nhất

- Độ phức tạp giải thuật: tất cả các thao tac so sánh trong while

- Với k là số lần gán (j := j\*b).

- Đặt c là số lần so sánh (j <= n).

+ Ban đầu j = 1.

+ Sau khi lặp lần 1, j = b => số lần gán = 2, số lần so sánh = 1

+ Sau khi lặp lần 2, j = b2 => số lần gán = 3, số lần so sánh = 2

+ Sau khi lặp lần 3, j = b3 => số lần gán = 4, số lần so sánh = 3

…

+ Sau khi lặp lần c, j = bc => số lần gán = k (= c + 1), số lần so sánh = c

+ Ở điều kiện cuối cùng: j <= n mà j = bc nên bc <= n thì:

* Cần thực hiện 1 lần so sánh nữa để không thực hiện bên trong thân while nên tổng số lần thực hiện so sánh là c + 1.
* Với c <= logb(n) (đk: c, logb(n) > 0) nên Tổng thao tác so sánh: c + 1 <= logb(n) + 1

Vậy độ phức tạp giải thuật là log(n).

7.

Đặt N = 2n

C(N) = 2CN/2 + N + 1

C(2n) = 2C2n-1 + 2n + 1

= 2(2C2n-2 + 2n-1 + 1) + 2n + 1

= 22. C2n-2 + 2n + 21 + 2n + 1

= 22. (2C2n-3+ 2n-2 + 1)+ 2.2n + 1 + 21

= 23. C2n-3 + 3.2n + 1 + 21+ 22

…

= 2n. C1 + n.2n + 2n-1 + 2n-2 + … + 21 + 1

= (n+1)2n – 1 = (log2(N) + 1).N – 1

Vậy độ phức tạp O(Nlog2(N)).