
Problem 1. *Tính độ phức tạp của các công thức truy hồi sau:*

1. $\begin{cases} T(1) = 4 \\ T(n) = 3T(n-1) \quad \forall x > 1 \end{cases}$

   Ta có:

   $T(n) = 3T(n-1)$

   $\leftrightarrow T(n-1) = 3T(n-2)$

   $\leftrightarrow T(n-2) = 3T(n-3)$

   $\leftrightarrow T(n) = 27T(n-3)$

   ...

   $T(1) = 4$

   $\leftrightarrow T(n) = 3^i \times T(n-i)$

   Với $i = n-1$, ta có:

   $T(n) = 3^{n-1} \times T(1)$

   $= 4 \times 3^{n-1}$

   $\rightarrow$ Độ phức tạp là $O(3^n)$

2. $\begin{cases} T(1) = 1 \\ T(n) = 2T(\frac{n}{2}) + \frac{n}{2} \quad \forall x > 1 \end{cases}$

   Theo định lý Master, ta có: $a = 2, b = 2, d = 1$

   $\rightarrow a = b^d$

   $\rightarrow$ Độ phức tạp là $O(n \log(n))$

3. $\begin{cases} T(1) = 1 \\ T(n) = 7T(\frac{n}{4}) + n^2 \quad \forall x > 1 \end{cases}$

   Theo định lý Master, ta có: $a = 7, b = 4, d = 2$

   $\rightarrow a < b^d$

   $\rightarrow$ Độ phức tạp là $O(n^2)$

Problem 2. *Cho đoạn code Python sau:*

```python
def Search(val, left = 0, right = len(b) - 1):
    if left > right:
        return -1
    mid = (left + right) // 2
    if b[mid] == val:
        return mid
    elif b[mid] > val:
        return Search(val, left, mid - 1)
    else:
        return Search(val, mid + 1, right)
```

Trong đó, b là mảng chứa N phần tử đã được sắp xếp tăng dần.
Thực hiện những yêu cầu sau:

1. *Cho biết đoạn code trên đang làm gì và xuất ra gì ?*
2. *Xác định phần cơ sở và phần đệ quy.*
3. *Lập công thức truy hồi và tính toán độ phức tạp của đoạn code trên.*

Solution.

1. Đoạn code tìm phần tử có giá trị val trong mảng b đã được sắp xếp tăng dần. Đoạn code xuất ra chỉ số của phần tử thỏa mãn nếu tìm thấy và xuất ra -1 nếu không tìm thấy.
2. Phần cơ sở: if left $>$ right: return -1 và if b[mid] == val: return mid

   Phần đệ quy: Search(val, left, mid – 1) và Search(val, mid + 1, right)
3. Công thức truy hồi:

   $$\begin{cases} T(0) = 0 \\ T(n) = T(\frac{n}{2}) + 1 \end{cases}$$

   Theo định lí Master, ta có: $a = 1, b = 2, d = 0$

   $\rightarrow a = b^d$

   $\rightarrow$ Độ phức tạp là $O(logn)$

□

Problem 3. *Có 2 máy in, mỗi máy in 1 mặt giấy mất 1 phút. Người ta có N tờ giấy cần được in 2 mặt. 2 máy in này có thể hoạt động song song với nhau.*

Tính thời gian ít nhất để in hoàn tất N tờ giấy. Xem xét một thuật toán đệ quy như sau:

- *Nếu $n \leq 2$, in 1 hoặc 2 tờ giấy cùng lúc vào mỗi mặt trên 1 hoặc 2 máy in.*
- *Nếu $n > 2$, in 2 mặt của 2 tờ giấy bất kỳ cùng lúc trên 2 máy in và tiếp tục cho đến hết $n - 2$ tờ giấy còn lại.*

1. *Lập công thức truy hồi của thuật toán trên cho n tờ giấy.*
2. *Giải thích vì sao thuật toán trên không cho ra thời gian tối ưu (ít nhất) để in n tờ giấy trên 2 máy in.*
3. *Đưa ra thuật toán đệ quy in ra thời gian ngắn nhất để n tờ giấy trên 2 máy in, Lập công thức truy hồi của thuật toán vừa đưa ra.*

Solution.

1. Công thức đệ quy tính thời gian:

$$\begin{cases} F(1) & = F(2) = 2 \\ F(n) & = F(n-2) + 2, \forall n > 2 \end{cases}$$

   Từ đó ta có công thức truy hồi tính độ phức tạp:

$$\begin{cases} T(1) & = T(2) = 0 \\ T(n) & = T(n-2) + 1, \forall n > 2 \end{cases}$$

2. Với $N = 3$, theo thuật toán trên ta có $F(N) = 4$.

   Tuy nhiên ta có thể làm như sau: in cùng lúc mặt trước của tờ 1 với mặt sau của tờ 2, mặt trước của tờ 2 với mặt sau của tờ 3, mặt trước của tờ 3 với mặt sau của tờ 1.

   Làm như trên chỉ mất $F_2(n) = 3$

3. Nếu $n \leq 2$, in 1 hoặc 2 tờ giấy cùng lúc vào mỗi mặt trên 1 hoặc 2 máy in

   Nếu $n = 3$, làm như câu b

   Nếu $n > 3$, in 2 mặt của 2 tờ giấy bất kì cùng lúc trên 2 máy in và tiếp tục cho đến hết n – 2 tờ giấy còn lại

   Công thức đệ quy tính thời gian:

$$\begin{cases} F_2(1) & = F_2(2) = 2 \\ F_2(3) & = 3 \\ F_2(n) & = (n-2) + 2, \forall n > 3 \end{cases}$$

Công thức truy hồi tính độ phức tạp

$$\begin{cases} T_2(1) & = T_2(2) = T_2(3) = 0 \\ T_2(n) & = T(n-2) + 1, \forall n > 3 \end{cases}$$

□