TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KHOA HỌC MÁY TÍNH

BÀI TẬP MÔN
PHÂN TÍCH VÀ THIẾT KẾ THUẬT TOÁN

HOMEWORK 03:

ĐỘ PHỨC TẠP VÀ CÁC KÝ HIỆU TIỆM CẬN

GV hướng dẫn: Huỳnh Thị Thanh Thương

Nhóm sinh viên thực hiện:

Họ và tên	***MSSV***	***Mã lớp***
1. Trần Đình Khánh Đăng	**22520195**	**CS112.O22**
2. Lê Minh Nhựt	**22521060**	**CS112.O22**
3. Lê Cảnh Nhật	**22521016**	**CS112.O22**
4. Nguyễn Hùng Phát	**22521074**	**CS112.O22**

TP. Hồ Chí Minh, ngày 28 tháng 4 năm 2024

BÀI 1:

Bảng giá trị n lớn nhất ứng với hàm f(n) và thời gian t

	1 second	1 minute	1 hour	1 day	1 month	1 year	1 century
$\lg n$	2^{10^6}	$2^{6.10^7}$	$2^{3,6.10^9}$	$2^{8,64.10^{10}}$	$2^{2,59.10^{12}}$	$2^{3,15.10^{13}}$	$2^{3,15.10^{15}}$
$\sqrt{n}$	10^{12}	$3,6.10^{15}$	$1,3.10^{19}$	$7,46.10^{21}$	$6,27.10^{24}$	$9,95.10^{26}$	$9,95.10^{30}$
n	10^6	6.10^7	$3,6.10^9$	$8,64.10^{10}$	$2,59.10^{12}$	$3,15.10^{13}$	$3,15.10^{15}$
n $\lg n$	$6,27.10^4$	$2,8.10^6$	$1,33.10^8$	$2,76.10^9$	$7,19.10^{10}$	$7,98.10^{11}$	$6,86.10^{13}$
n^2	1000	7745	60000	293938	1609968	5615692	56156922
n^3	100	391	1532	4420	13736	31593	146645
2^n	19	25	31	36	41	44	51
n!	9	11	12	13	15	16	17

- Chương trình chạy bằng Python:

```
import math
'''Định nghĩa hàm log_2(n)'''
def log2(n):
  return math.log(n) / math.log(2)

'''Danh sách các hàm phức tạp tính toán thời gian chạy'''
complexities = [lambda n: math.sqrt(n),   # f(n)=sqrt()
                lambda n: n,              # f(n)=n
                lambda n: n * log2(n),   # f(n)=log_2(n)
                lambda n: n ** 2,        # f(n)=n^2
                lambda n: n ** 3,        # f(n)=n^3
                lambda n: 2 ** n,        # f(n)=2^n
                lambda n: math.factorial(n)]  # f(n)=n!

'''Giới hạn tối đa cho từng hàm phức tạp'''
max_bound = [1e40, 1e20, 1e20, 1e10, 1e10, 100, 100]

'''Thời gian thực thi tương ứng, với số phép tính máy tính có thể
tính toán trong 1 giây là 10^6'''
times = [1000000,                              # 1 second
         1000000 * 60,                         # 1 minute
         1000000 * 60 * 60,                    # 1 hour
         1000000 * 60 * 60 * 24,               # 1 day
         1000000 * 60 * 60 * 24 * 30,          # 1 month
         1000000 * 60 * 60 * 24 * 365,         # 1 year
         1000000 * 60 * 60 * 24 * 365 * 100] # 1 century

'''Hàm định dạng thời gian'''
def format_time(t):
  '''Tính số mũ của 10 cho giá trị thời gian t'''
  exp = int(math.log10(t))
  '''Tính hệ số cho giá trị thời gian t, làm tròn tới chữ số
  thập phân thứ 2'''
  coeff = round(t / (10 ** exp), 2)
```

```
  '''Kiểm tra nếu hệ số là 1.0'''
  if coeff == 1.0:
    '''Trả về chuỗi biểu diễn 10 mũ exp'''
    return '10^{}'.format(exp)
  else:
    '''Trả về chuỗi biểu diễn hệ số nhân với 10 mũ exp'''
    return '{}*10^{}'.format(coeff, exp)

print('1 giây | 1 phút | 1 giờ | 1 ngày | 1 tháng | 1 năm | 1 thế kỷ')
'''In ra các giá trị n lớn nhất có thể thực thi của hàm lgn'''
print('lgn: 2^(', end='')
print(') | 2^('.join(map(format_time, times)), end='')
print(')')
'''Do giá trị của n với trường hợp lgn quá lớn nên
không thể tính toán bằng máy tính theo phương pháp bên dưới
vì vậy em dùng cách trên để có thể biểu diễn kết quả'''

'''Duyệt qua từng hàm phức tạp tính toán thời gian chạy'''
for k in range(len(complexities)):
  '''Lấy hàm phức tạp tính toán thời gian chạy thứ k từ
  danh sách complexities'''
  c = complexities[k]
  '''Tạo một danh sách để lưu trữ giá trị lớn nhất của n cho từng
  thời gian tương ứng'''
  values = []
  '''Duyệt qua từng giá trị thời gian trong danh sách times'''
  for t in times:
    '''Khởi tạo biến left và right cho thuật toán tìm kiếm nhị phân'''
    left, right = 0, int(max_bound[k])
    '''Khởi tạo biến max_n để lưu trữ giá trị lớn nhất của n
    thỏa mãn thời gian t'''
    max_n = 0
    '''Thực hiện thuật toán tìm kiếm nhị phân để tìm giá trị lớn nhất
    của n thỏa mãn thời gian t'''
    while left <= right:
      mid = (left + right) // 2
      '''Tính toán thời gian thực thi của hàm c với đầu vào là mid'''
      value = c(mid)
      '''Nếu thời gian thực thi không xác định hoặc vượt
      quá thời gian t'''
      if value == float('inf') or value > t:
        '''Giảm giá trị right để thuật toán tiếp tục tìm kiếm
        ở phía bên trái'''
        right = mid - 1
      else:
        '''Tăng giá trị left để thuật toán tiếp tục tìm kiếm
        ở phía bên phải'''
        left = mid + 1
        '''Cập nhật giá trị lớn nhất của n thỏa mãn thời gian t'''
        max_n = max(max_n, mid)
    '''Sau khi tìm được giá trị lớn nhất của n cho thời gian t,
    thêm giá trị này vào danh sách values'''
```

```
    values.append(max_n)

  '''In ra giá trị n tương ứng cho từng thời gian và hàm phức tạp'''
  if k < 3:
    if (k==0):
      print('sqrt(n):', end=' ')
    if (k==1):
      print('n:', end=' ')
    if (k==2):
      print('nlgn:', end=' ')
    '''In ra giá trị lớn nhất của n cho mỗi thời gian tương ứng
    với hàm phức tạp theo định dạng {}.10^{}'''
    print(' | '.join(map(lambda v: format_time(v), values)))
  else:
    if (k==3):
      print('n^2:', end=' ')
    if (k==4):
      print('n^3:', end=' ')
    if (k==5):
      print('2^n:', end=' ')
    if (k==6):
      print('n!:', end=' ')
    '''In ra giá trị lớn nhất của n cho mỗi thời gian tương ứng
    với hàm phức tạp, làm tròn xuống'''
    print(' | '.join(map(lambda v: str(int(math.floor(v))), values)))
```

Kết quả chạy chương trình:

```
1 giây | 1 phút | 1 giờ | 1 ngày | 1 tháng | 1 năm | 1 thế kỷ
lgn: 2^(10^6) | 2^(6.0*10^7) | 2^(3.6*10^9) | 2^(8.64*10^10) | 2^(2.59*10^12) | 2^(3.15*10^13) | 2^(3.15*10^15)
sqrt(n): 10^12 | 3.6*10^15 | 1.3*10^19 | 7.46*10^21 | 6.72*10^24 | 9.95*10^26 | 9.95*10^30
n: 10^6 | 6.0*10^7 | 3.6*10^9 | 8.64*10^10 | 2.59*10^12 | 3.15*10^13 | 3.15*10^15
nlgn: 6.27*10^4 | 2.8*10^6 | 1.33*10^8 | 2.76*10^9 | 7.19*10^10 | 7.98*10^11 | 6.86*10^13
n^2: 1000 | 7745 | 60000 | 293938 | 1609968 | 5615692 | 56156922
n^3: 100 | 391 | 1532 | 4420 | 13736 | 31593 | 146645
2^n: 19 | 25 | 31 | 36 | 41 | 44 | 51
n!: 9 | 11 | 12 | 13 | 15 | 16 | 17
```

- Giải thích thêm bằng lời:
+ Ta có số phép mà máy tính có thể thực trong 1 giây là 10^6.
+ Vậy ta cần giải n với hàm: $f(n) = 10^6.t$
+ Ví dụ: Tính n với một số hàm f(n) khi t = 1s
$n = 10^6$
$\lg n = 10^6 => n = 2^{10^6}$
$\sqrt{n} = 10^6 => n = (10^6)^2 = 10^{12}$
$n^2 = 10^6 \Leftrightarrow n^2 = (10^3)^2 => n = 10^3$
+Với các hàm f(n) và t khác ta cũng làm tương tự, riêng 2 hàm $f(n) = n$ và $f(n) = n!$ không phải lúc nào cũng rõ ràng để tính toán nên ta cần đến đoạn code bên trên.
+ Đoạn code trên có ý tưởng chính là duyệt qua mỗi hàm phức tạp và mỗi giới hạn thời gian, sau đó thực hiện thuật toán tìm kiếm nhị phân để tìm giá trị lớn nhất của n. Với hàm $\lg n$ ta có thể thấy ở ví dụ trên n tìm được rất lớn nên không

thể tìm kiếm nhị phân vì vậy trong đoạn code em sẽ biểu diễn nó với dạng $2^{10^6.t}$

BÀI 2: Sắp xếp tăng dần "theo Big O nhỏ nhất"

Ta có: $\log n \in O(n^c)$ với c có thể rất bé và $c > 0$ (dùng cho toàn bộ bài 2)

1. **Group 1:**

$$f_1(n) = \binom{n}{100} = \frac{n!}{100!(n-100)!}$$
$$= \frac{(n-100)!(n-99)(n-98)...n}{100!(n-100)!} = \frac{(n-99)(n-98)...n}{100!} = O(n^{100})$$
$$f_2(n) = n^{100} = O(n^{100})$$
$$f_3(n) = \frac{1}{n} = n^{-1} = O(n^{-1})$$
$$f_4(n) = 10^{1000}n = O(n)$$
$$f_5(n) = n\log n = O(n.n^c) = O(n^{1+c})$$

Vậy thứ tự tăng dần theo Big-O nhỏ nhất của các hàm số là:
$f_3(n) < f_4(n) < f_5(n) < f_1(n) = f_2(n)$

2. **Group 2:**

$$f_1(n) = 2^{2^{1000000}} = O(1)$$
$$f_2(n) = 2^{100000n} = O((2^{100000})^n)$$
$$f_3(n) = \binom{n}{2} = \frac{n!}{2!(n-2)!} = \frac{(n-2)!(n-1)n}{2!(n-2)!} = \frac{(n-1)n}{2!} = O(n^2)$$
$$f_4(n) = n\sqrt{n} = O(n^{3/2})$$

Vậy thứ tự tăng dần theo Big-O nhỏ nhất của các hàm số là:
$f_1(n) < f_4(n) < f_3(n) < f_2(n)$

3. **Group 3:**

$$f_1(n) = n^{\sqrt{n}} = (2^{logn})^{\sqrt{n}} = 2^{\sqrt{n}.logn} = 2^{O(n^{0.5}.n^c)} = 2^{O(n^{0.5+c})}$$
$$f_2(n) = 2^n = 2^{O(n)}$$
$$f_3(n) = n^{10}.2^{n/2} = (2^{logn})^{10}.2^{n/2} = 2^{10logn+n/2} = 2^{O(n)}$$
$$f_4(n) = \sum_{i=1}^{n}(i+1) = \sum_{i=1}^{n} i + \sum_{i=1}^{n} 1 = \frac{n(n+1)}{2} + n = \frac{n^2+3n}{2}$$
$$\approx n^2 = (2^{\log n})^2 = 2^{2\log n} = 2^{O(n^c)}$$

Vậy thứ tự tăng dần theo Big-O nhỏ nhất của các hàm số là:
$f_4(n) < f_1(n) < f_2(n) = f_3(n)$

4. **Group 4:**

$$f_6(n) = n^{\sqrt{n}} = (2^{logn})^{\sqrt{n}} = 2^{\sqrt{n}.logn} = 2^{O(n^{0.5}.n^c)} = 2^{O(n^{0.5+c})}$$
$$f_7(n) = \pi^n = (2^{log\pi})^n = 2^{log\pi.n} = 2^{O(n)}$$
$$f_8(n) = 2^{n^4} = 2^{O(n^4)}$$
$$f_9(n) = n^{4\log n} = (2^{logn})^{4logn} = 2^{4logn.logn} = 2^{O(n^{2c})}$$

Vậy thứ tự tăng dần theo Big-O nhỏ nhất của các hàm số là:
$f_9(n) < f_6(n) < f_7(n) < f_8(n)$

5. **Group 5:**
$f_6(n) = n^{\sqrt{n}} = (2^{logn})^{\sqrt{n}} = 2^{\sqrt{n}.logn} = 2^{O(n^{0.5}.n^c)} = 2^{O(n^{0.5+c})}$
$f_7(n) = n^{logn} = (2^{logn})^{logn} = 2^{logn.logn} = 2^{O(n^{2c})}$
$f_8(n) = 2^{n/2} = 2^{O(n)}$
$f_9(n) = 3^{\sqrt{n}} = (2^{\log_2 3})^{n^{0.5}} = 2^{log_2 3.n^{0.5}} = 2^{O(n^{0.5})}$
$f_{10}(n) = 4^{n^{1/4}} = (2^2)^{n^{0.25}} = 2^{2.n^{0.25}} = 2^{O(n^{0.25})}$
Ta có: $\log n \in O(n^c)$ với c có thể rất bé và $c > 0$
Vậy thứ tự tăng dần theo Big-O nhỏ nhất của các hàm số là:
$f_7(n) < f_{10}(n) < f_9(n) < f_6(n) < f_8(n)$

6. **Group 6:**
$f_1(n) = n^{0.999999}\log n = O(n^{0.999999+c})$
$f_2(n) = 10000000n = O(n)$
$f_3(n) = 1.000001^n = O(2^n)$
$f_4(n) = n^2 = O(n^2)$
Vậy thứ tự tăng dần theo Big-O nhỏ nhất của các hàm số là:
$f_1(n) < f_2(n) < f_4(n) < f_3(n)$

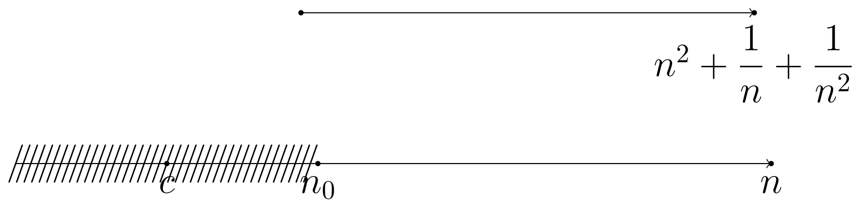
7. **Group 7:**
$f_1(n) = n^{\pi} = O(n^{\pi})$
$f_2(n) = \pi^n = (2^{log_2 \pi})^n = 2^{nlog_2 \pi} = 2^{O(n)}$
$$f_3(n) = \binom{n}{5} = \frac{n!}{(n-5)!5!} = \frac{n(n-1)(n-2)(n-3)(n-4)}{5!} = O(n^5)$$
$f_4(n) = \sqrt{2^{\sqrt{n}}} = (2^{\sqrt{n}})^{0.5} = 2^{0.5\sqrt{n}} = 2^{O(n^{0.5})}$
$$f_5(n) = \binom{n}{n-4} = \frac{n!}{(n-4)!4!} = \frac{(n-3)(n-2)(n-1)n}{4!} = O(n^4)$$
$f_6(n) = 2^{\log^4 n} = 2^{O(n^{4c})}$
$f_7(n) = n^{5(\log n)^2} = (2^{logn})^{5(logn)^2} = 2^{5(logn)^2.logn} = 2^{O(n^{2c}.n^c)} = 2^{O(n^{3c})}$
$$f_8(n) = n^4\binom{n}{4} = n^4\frac{n!}{(n-4)!4!} = n^4\frac{(n-3)(n-2)(n-1)n}{4!} = O(n^8)$$
$$\Rightarrow \begin{cases} f_1(n) < f_5(n) < f_3(n) < f_8(n) \\ f_7(n) < f_6(n) < f_4(n) < f_2(n) \end{cases}$$
Ta có:
$f_8(n) \approx n^8 = (2^{\log n})^8 = 2^{8\log n} = 2^{O(n^c)}$
$f_7(n) = 2^{O(n^{3c})}$
$\Rightarrow f_8(n) < f_7(n)$
Vậy thứ tự tăng dần theo Big-O nhỏ nhất của các hàm số là:
$f_1(n) < f_5(n) < f_3(n) < f_8(n) < f_7(n) < f_6(n) < f_4(n) < f_2(n)$

BÀI 3: Chứng minh, dùng định nghĩa của các ký hiệu tiệm cận (không dùng lim)

a. Đề bài
$n^4 + n + 1 \notin O(n^2)$
Lời giải

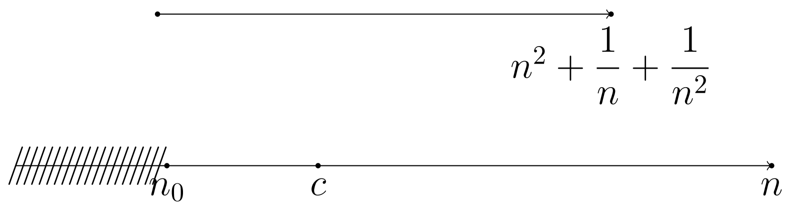
- Giả sử: $n^4 + n + 1 \in O(n^2)$
  $\Rightarrow \exists c \in \mathbb{R}^+, n_0 \in \mathbb{N}$ sao cho : $n^4 + n + 1 \leq c.n^2, \forall n \geq n_0$
  $\Leftrightarrow n^2 + \dfrac{1}{n} + \dfrac{1}{n^2} \leq c, \forall n \geq n_0$

- TH1: $n_0 \geq c$



  Trường hợp này mâu thuẫn với giải thiết vì $\forall n \geq n_0, n^2 + \dfrac{1}{n} + \dfrac{1}{n^2} > c$

- TH2: $n_0 < c$



  Trường hợp này mâu thuẫn với giải thiết vì $\exists n \geq n_0, n^2 + \dfrac{1}{n} + \dfrac{1}{n^2} > c$

- Suy ra giả sử ban đầu là sai

- Vậy: $n^4 + n + 1 \notin O(n^2)$

b. Đề bài
$O(C.f(n)) = O(f(n))$với C là hằng số
Lời giải

- Xét 1 hàm $d_1(n)$ bất kì $\in O(C.f(n))$
  $\Rightarrow \exists c_1 \in \mathbb{R}^+, n_1 \in \mathbb{N}$ sao cho: $d_1(n) \leq c_1 C.f(n), \forall n \geq n_1$
  $\Rightarrow \exists c_2 = c_1.C \in \mathbb{R}^+, n_2 = n_1 \in \mathbb{N}$ sao cho: $d_1(n) \leq c_2.f(n), \forall n \geq n_2$
  $\Rightarrow d_1(n) \in O(f(n))$
  Vậy: $O(C.f(n)) \subseteq O(f(n))(1)$

- Xét 1 hàm $d_2(n)$ bất kì $\in O(f(n))$
  $\Rightarrow \exists c_3 \in \mathbb{R}^+, n_3 \in \mathbb{N}$ sao cho: $d_2(n) \leq c_3.f(n), \forall n \geq n_3$

$\Rightarrow \exists c_4 = \dfrac{c_3}{C} \in \mathbb{R}^+, n_4 = n_3 \in \mathbb{N}$ sao cho: $d_2(n) \leq c_4.C.f(n), \forall n \geq n_4$
$\Rightarrow d_2(n) \in O(C.f(n))$
Vậy: $O(f(n)) \subseteq O(C.f(n))(2)$
Từ (1) và (2) $\Rightarrow O(C.f(n)) = O(f(n))$

c. Đề bài
Nếu$f(n) \in O(g(n))$ và $g(n) \in O(h(n))$ thì $f(n) \in O(h(n))$
Lời giải

- Ta có: $f(n) \in O(g(n))$
  $\Rightarrow \exists c_1 \in \mathbb{R}^+, n_1 \in \mathbb{N}$ sao cho: $f(n) \leq c_1 g(n), \forall n \geq n_1(1)$
- Ta có: $g(n) \in O(h(n))$
  $\Rightarrow \exists c_2 \in \mathbb{R}^+, n_2 \in \mathbb{N}$ sao cho:
  $g(n) \leq c_2 h(n), \forall n \geq n_2 \Leftrightarrow c_1.g(n) \leq c_1.c_2.h(n), \forall n \geq n_2(2)$
- Từ (1) và (2) suy ra: $f(n) \leq c_1.c_2.h(n), \forall n \geq n_1, n \geq n_2$
  $\Rightarrow \exists\, d = c_1.c_2 \in \mathbb{R}^+, n_0 = \max\{n_1, n_2\} \in \mathbb{N}$ sao cho:
  $f(n) \leq d.h(n), \forall n \geq n_0$
  $\Rightarrow f(n) \in O(h(n))$ (theo định nghĩa Big-O ta có điều phải chứng minh)

d. Đề bài
$max\{f(n), g(n)\} = \Theta(f(n) + g(n))$
Lời giải

- Ta cần chứng minh $\exists c_1, c_2 \in \mathbb{R}^+, n_0 \in \mathbb{N}$ sao cho:
  $c_1(f(n) + g(n)) \leq max\{(f(n), g(n)\} \leq c_2(f(n) + g(n)), \forall n \geq n_0$
- Ta có: $\begin{cases} f(n) \leq max\{f(n), g(n)\} \\ g(n) \leq max\{f(n), g(n)\} \end{cases}$
  $\Rightarrow max\{f(n), g(n)\} \leq f(n) + g(n) \leq 2max\{f(n), g(n)\}$
  $\Leftrightarrow \begin{cases} max\{f(n), g(n)\} \leq f(n) + g(n) \\ f(n) + g(n) \leq 2max\{f(n), g(n)\} \end{cases}$
  $\Leftrightarrow \begin{cases} max\{f(n), g(n)\} \leq f(n) + g(n) \\ \dfrac{1}{2}(f(n) + g(n)) \leq max\{f(n), g(n)\} \end{cases}$
- $\exists c_1 = \dfrac{1}{2}, c_2 = 1 \in \mathbb{R}^+, n_0 = 1 \in \mathbb{N}$ sao cho:
  $\dfrac{1}{2}(f(n) + g(n)) \leq max\{f(n), g(n)\} \leq f(n) + g(n), \forall n \geq 1$
- Theo định nghĩa Big-$\Theta \Rightarrow max\{f(n), g(n)\} = \Theta(f(n) + g(n))$ (đpcm)

e. Đề bài
$g(n) \in O(h(n)) \Rightarrow O(g(n)) \subseteq O(h(n))$
Lời giải

- Ta có: $g(n) \in O(h(n))$
  $\Rightarrow \exists\ c_0 \in \mathbb{R}^+, n_0 \in \mathbb{N}$ sao cho:
  $g(n) \leq c_0.h(n), \forall n \geq n_0$ (1)

- Chứng minh: $O(g(n)) \subseteq O(h(n))$

- Xét 1 hàm $f(n)$ bất kỳ $\in O(g(n))$
  $\Rightarrow \exists\ c_1 \in \mathbb{R}^+, n_1 \in \mathbb{N}$ sao cho:
  $f(n) \leq c_1.g(n), \forall n \geq n_1$ (2)

- Từ (1) suy ra: $c_1.g(n) \leq c_0.c_1.h(n), \forall n \geq n_0$ (3)

- Từ (2) và (3) suy ra:
  $f(n) \leq c_0 c_1.h(n), \forall n \geq n_0, n \geq n_1$
  $\Rightarrow \exists\ d = c_0.c_1 \in \mathbb{R}^+, n_2 = \max\{n_0, n_1\} \in \mathbb{N}$ sao cho:
  $f(n) \leq d.h(n), \forall n \geq n_2$
  $\Rightarrow f(n) \in O(h(n))$

- Vậy: $O(g(n)) \subseteq O(h(n))$

f. Đề bài

$\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$

Lời giải

- Xét 1 hàm $f(n)$ bất kì $\in \Theta(g(n))$
  $\Rightarrow \exists\ c_1, c_2 \in \mathbb{R}^+, n_0 \in \mathbb{N}$ sao cho:
  $c_1.g(n) \leq f(n) \leq c_2.g(n), \forall n \geq n_0$
  Hay $\begin{cases} \exists\ c_1 \in \mathbb{R}^+, n_0 \in \mathbb{N} \text{ sao cho: } c_1.g(n) \leq f(n), \forall n \geq n_0 \\ \exists\ c_2 \in \mathbb{R}^+, n_0 \in \mathbb{N} \text{ sao cho: } f(n) \leq c_2.g(n), \forall n \geq n_0 \end{cases}$
  $\Rightarrow \begin{cases} f(n) \in \Omega(g(n)) \\ f(n) \in O(g(n)) \end{cases} \quad \Rightarrow f(n) \in O(g(n)) \cap \Omega(g(n))$

- Vậy: $\Theta(g(n)) \subset O(g(n)) \cap \Omega(g(n))(1)$

- Xét 1 hàm $h(n)$ bất kỳ $\in O(g(n)) \cap \Omega(g(n))$
  $\Rightarrow \begin{cases} h(n) \in O(g(n)) \\ h(n) \in \Omega(g(n)) \end{cases}$
  $\Rightarrow \begin{cases} \exists c_3 \in \mathbb{R}^+, n_3 \in \mathbb{N} \text{ sao cho: } h(n) \leq c_3.g(n), \forall n \geq n_3 \\ \exists c_4 \in \mathbb{R}^+, n_4 \in \mathbb{N} \text{ sao cho: } h(n) \geq c_4.g(n), \forall n \geq n_4 \end{cases}$
  $\Rightarrow \exists c_3, c_4 \in \mathbb{R}^+, n_5 = \max\{n_3, n_4\} \in \mathbb{N}$ sao cho:
  $c_4.g(n) \leq h(n) \leq c_3.g(n), \forall n \geq n_5$
  $\Rightarrow h(n) \in \Theta(g(n))$

- Vậy: $O(g(n)) \cap \Omega(g(n)) \subset \Theta(g(n))(2)$

- Từ (1) và (2) suy ra: $\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$ (đpcm)

g. Đề bài

$n + n^2 O(\ln n) = O(n^2 \ln n)$

Lời giải

- Xét 1 hàm $f(n)$ bất kỳ $\in O(\ln n)$
  $\Rightarrow \exists c_1 \in \mathbb{R}^+, n_1 \in \mathbb{N}$ sao cho: $f(n) \leq c_1. \ln n, \forall n \geq n_1$
  $\Leftrightarrow n^2 f(n) \leq c_1.n^2 \ln n, \forall n \geq n_1$ (1)
- Mà: $n \leq n^2 \ln n, \forall n \geq e$ $(e \approx 2.71828$ và $\ln e = 1)$ (2)
- Cộng (1) và (2), ta có:
  $n + n^2 f(n) \leq (c_1 + 1).n^2 \ln n, \forall n \geq n_1, n \geq e$
  $\Rightarrow n + n^2 O(\ln n) \leq (c_1 + 1).n^2 \ln n, \forall n \geq n_1, n \geq e$
  $\Rightarrow \exists d = c_1 + 1 \in \mathbb{R}^+, n_2 = \max\{n_1, e\} \in \mathbb{N}$ sao cho:
  $n + n^2 O(\ln n) \leq d.n^2 \ln n, \forall n \geq n_2$
- Theo định nghĩa Big - O suy ra: $n + n^2 O(\ln n) = O(n^2 \ln n)$

Bài 4: Các khẳng định là đúng hay sai? Vì sao?

a. Đề bài

Nếu $f(n) = \Theta(g(n))$ và $g(n) = \Theta(h(n)),$ thì $h(n) = \Theta(f(n))$

Lời giải

- Ta có: $f(n) = \Theta(g(n))$
  $\Rightarrow \exists\, c_1, c_2 \in \mathbb{R}^+, n_1 \in \mathbb{N}$ sao cho:
  $c_1.g(n) \leq f(n) \leq c_2.g(n), \forall n \geq n_1$ (1)

- Ta có: $g(n) = \Theta(h(n))$
  $\Rightarrow \exists\, c_3, c_4 \in \mathbb{R}^+, n_2 \in \mathbb{N}$ sao cho:
  $c_3.h(n) \leq g(n) \leq c_4.h(n), \forall n \geq n_2$ (2)

- Từ (1) và (2), ta được:
  $c_3c_1.h(n) \leq c_1.g(n) \leq f(n), \forall n \geq \max\{n_1, n_2\}$
  $\Leftrightarrow h(n) \leq \dfrac{1}{c_3c_1}.f(n), \forall n \geq \max\{n_1, n_2\}$ (3)
  $f(n) \leq c_2.g(n) \leq c_2c_4.h(n), \forall n \geq \max\{n_1, n_2\}$
  $\Leftrightarrow \dfrac{1}{c_2c_4}.f(n) \leq h(n), \forall n \geq \max\{n_1, n_2\}$ (4)

- Từ (3) và (4), ta được:
  $\dfrac{1}{c_2c_4}f(n) \leq h(n) \leq \dfrac{1}{c_1c_3}f(n), \forall n \geq \max\{n_1, n_2\}$
  $\Rightarrow \exists\, d_1 = \dfrac{1}{c_2c_4}, d_2 = \dfrac{1}{c_1c_3} \in \mathbb{R}^+, n_3 = \max\{n_1, n_2\} \in \mathbb{N}$ sao cho:
  $d_1.f(n) \leq h(n) \leq d_2.f(n), \forall n \geq n_3$
  $\Rightarrow h(n) = \Theta(f(n))$. (theo định nghĩa Big - Θ)

- Vậy khẳng định trên là đúng

b. Đề bài

Nếu$f(n) = O(g(n))$ và $g(n) = O(f(n))$ thì $f(n) = g(n)$

Lời giải

- Khẳng định trên là sai, vì tồn tại hai hàm số:
  $f(n) = n + 2$
  $g(n) = 2n + 1$

- Ta thấy hai hàm số này thỏa mãn các điều kiện:
  $f(n) = O(g(n)) = O(2n + 1) = O(n)$
  $g(n) = O(f(n)) = O(n + 2) = O(n)$
  Nhưng hai hàm số $f(n)$ và $g(n)$ không bằng nhau

- Vậy khẳng định trên là sai

c. Đề bài

$f(n) + O(f(n)) = \Theta(f(n))$

Lời giải

- Xét 1 hàm bất kỳ $g(n) \in O(f(n))$
  $\Rightarrow \exists c \in \mathbb{R}^+, n_0 \in \mathbb{N}$ sao cho: $g(n) \leq c.f(n), \forall n \geq n_0$
  $\Rightarrow f(n) + g(n) \leq f(n) + c.f(n), \forall n \geq n_0$
  $\Leftrightarrow f(n) + g(n) \leq (c+1)f(n), \forall n \geq n_0(1)$
- Mà $f(n) \leq f(n) + g(n), \forall n \geq n_0(2)$
- Chọn $c_1 = 1, c_2 = c + 1$ .Từ (1) và (2) suy ra:
  $c_1 f(n) \leq f(n) + g(n) \leq c_2 f(n), \forall n \geq n_0$
  $\Rightarrow f(n) + g(n) = \Theta(f(n))$
  $\Rightarrow f(n) + O(f(n)) = \Theta(f(n))$
- Vậy: khẳng định $f(n) + O(f(n)) = \Theta(f(n))$ là đúng.

d. Đề bài

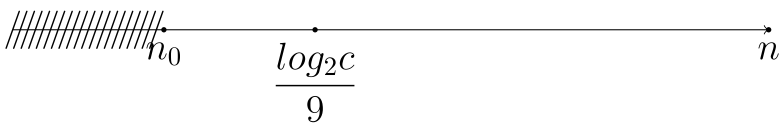
$2^{10n} = O(2^n)$

Lời giải

- Giả sử $2^{10n} = O(2^n)$
  $\Rightarrow \exists c \in \mathbb{R}^+$, $n_0 \in \mathbb{N}$ sao cho:
  $2^{10n} \leq c.2^n, \forall n \geq n_0$
  $\Leftrightarrow 2^{9n} \leq c, \forall n \geq n_0$
  $\Leftrightarrow n \leq \dfrac{log_2 c}{9}, \forall n \geq n_0$
- **TH1**: $n_0 > \dfrac{log_2 c}{9}$

Mâu thuẫn với điều kiện trên vì $\forall n \geq n_0, n > \dfrac{log_2 c}{9}$

- **TH2**: $n_0 \leq \dfrac{log_2 c}{9}$



Mâu thuẫn với điều kiện vì $\forall n \geq n_0$,vẫn tồn tại $n > \dfrac{log_2 c}{9}$

- Suy ra giả sử ban đầu là sai
- Vậy khẳng định $2^{10n} = O(2^n)$ là sai

e. Đề bài
$2^{n+10} = O(2^n)$
Lời giải

- Giả sử $2^{n+10} = O(2^n)$
  $\Rightarrow \exists c \in \mathbb{R}^+, n_0 \in \mathbb{N}$ sao cho : $2^{n+10} \leq c.2^n, \forall n \geq n_0$
- Chọn $n_0 = 1, c = 2^{10}$ ta có: $2^{n+10} \leq 2^{10}.2^n, \forall n \geq 1$ (đúng)
- Vậy khẳng định $2^{n+10} = O(2^n)$ là đúng

f. Đề bài
$log_a n = \Theta(log_b n)$
Lời giải

- Giả sử $log_a n = \Theta(log_b n)$
  $\Rightarrow \exists c_1, c_2 \in \mathbb{R}^+, n_0 \in \mathbb{N}$ sao cho :
  $c_1.log_b n \leq log_a n \leq c_2.log_b n, \forall n \geq n_0(1)$
- Chọn $c_1 = c_2 = \dfrac{1}{log_b a}, n_0 = 1$ thay vào (1) ta có:
  $\dfrac{1}{log_b a} log_b n \leq log_a n \leq \dfrac{1}{log_b a} log_b n, \forall n \geq 1$
  $\Leftrightarrow log_a n \leq log_a n \leq log_a n, \forall n \geq 1$ (đúng)
- Vậy khẳng định $log_a n = \Theta(log_b n)$ là đúng

Bài 5: Ước lượng nhanh độ phức tạp của giải thuật đệ quy dùng Định lý Master

1. **Đề bài**
   $T(n) = 3T(\frac{n}{2}) + n^2$
   Lời giải

   - Ta có:
     $a = 3, b = 2$
     $f(n) = n \in \Theta(n^2) \Rightarrow d = 2$
   - Vì $a < b^d$ $(3 < 2^2)$ nên áp dụng case 1 của dạng đơn giản
   - Vậy: $T(n) = \Theta(n^d) = \Theta(n^2)$

2. **Đề bài**
   $T(n) = 7T\left(\frac{n}{3}\right) + n^2$
   Lời giải

   - Ta có:
     $a = 7, b = 3$
     $f(n) = n^2 \in \Theta(n^2) \Rightarrow d = 2$
   - Vì $a < b^d (7 < 3^2)$ nên áp dụng case 1 của dạng đơn giản
   - Vậy: $T(n) \in \Theta(n^d) = \Theta(n^2)$

3. **Đề bài**
   $T(n) = 3T(\frac{n}{3}) + \frac{n}{2}$
   Lời giải

   - Ta có:
     $a = 3, b = 3$
     $f(n) = \frac{n}{2} \in \Theta(n) \Rightarrow d = 1$
   - Vì $a = b^d = 3$ nên áp dụng case 2 của dạng đơn giản
   - Vậy: $T(n) = \Theta(n^d.logn) = \Theta(n.logn)$

4. **Đề bài**
   $T(n) = 16T(\frac{n}{4}) + n$
   Lời giải

   - Ta có: $a = 16, b = 4$
     $f(n) = n \in \Theta(n^1) \Rightarrow d = 1$

- Vì $a > b^d$ $(16 > 4^1)$, nên áp dụng case 3 của dạng đơn giản
- Vậy: $T(n) = \Theta(n^{log_b a}) = \Theta(n^{log_4 16}) = \Theta(n^2)$

5. **Đề bài**
   $T(n) = 2T(\dfrac{n}{4}) + n^{0.51}$
   Lời giải

- Ta có: $a = 2, b = 4$
  $f(n) = n^{0.51} \in \Theta(n^{0.51}) \Rightarrow d = 0.51$
- Vì $a < b^d$ $(2 < 4^{0.51})$, nên áp dụng case 1 của dạng đơn giản
- Vậy: $T(n) = \Theta(n^d) = \Theta(n^{0.51})$

6. **Đề bài**
   $T(n) = 3T\left(\dfrac{n}{2}\right) + n$
   Lời giải

- Ta có: $a = 3, b = 2$
  $f(n) = n \in \Theta(n) \Rightarrow d = 1$
- Vì $a > b^d(3 > 2^1)$ nên áp dụng case 3 của dạng đơn giản
- Vậy: $T(n) \in \Theta(n^{\log_b a}) = \Theta(n^{\log_2 3})$

7. **Đề bài**
   $T(n) = 3T\left(\dfrac{n}{3}\right) + \sqrt{n}$
   Lời giải

- Ta có:
  $a = 3, b = 3$
  $f(n) = \sqrt{n} \in \Theta(n^{0.5}) \Rightarrow d = 0.5$
- Vì $a > b^d(3 > 3^{0.5})$ nên áp dụng case 3 của dạng đơn giản
- Vậy: $T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\log_3 3}) = \Theta(n)$

8. **Đề bài**
   $T(n) = 4T(\dfrac{n}{2}) + cn$
   Lời giải

- Ta có:
  $a = 4, b = 2$
  $f(n) = cn \in \Theta(n^1) \Rightarrow d = 1$
- Vì $a > b^d$ $(4 > 2^1)$ nên áp dụng case 3 của dạng đơn giản

- Vậy: $T(n) = \Theta(n^{log_b a}) = \Theta(n^{log_2 4}) = \Theta(n^2)$

9. **Đề bài**
   $T(n) = 4T(\frac{n}{4}) + 5n$
   Lời giải

- Ta có:
  $a = 4, b = 4$
  $f(n) = 5n \in \Theta(n^1) \Rightarrow d = 1$
- Vì $a = b^d$ $(4 = 4^1)$ nên áp dụng case 2 của dạng đơn giản
- Vậy: $T(n) = \Theta(n^d \log n) = \Theta(n \log n)$

10. **Đề bài**
    $T(n) = 5T\left(\frac{n}{4}\right) + 4n$
    Lời giải

- Ta có:
  $a = 5, b = 4$
  $f(n) = 4n \in \Theta(n) \Rightarrow d = 1$
- Vì $a > b^d (5 > 4^1)$ nên áp dụng case 3 của dạng đơn giản
- Vậy: $T(n) \in \Theta(n^{\log_b a}) = \Theta(n^{\log_4 5})$

11. **Đề bài**
    $T(n) = 4T\left(\frac{n}{5}\right) + 5n$
    Lời giải

- Ta có:
  $a = 4, b = 5$
  $f(n) = 5n \in \Theta(n) \Rightarrow d = 1$
- Vì $a < b^d (4 < 5^1)$ nên áp dụng case 1 của dạng đơn giản
- Vậy: $T(n) = \Theta(n^d) = \Theta(n)$

12. **Đề bài**
    $T(n) = 25T(\frac{n}{5}) + n^2$
    Lời giải

- Ta có:
  $a = 25, b = 5$
  $f(n) = n^2 \in \Theta(n^2) \Rightarrow d = 2$
- Vì $a = b^d (25 = 5^2)$ nên áp dụng case 2 của dạng đơn giản

- Vậy: $T(n) = \Theta(n^d logn) = \Theta(n^2 logn)$

13. Đề bài
$T(n) = 10T(\frac{n}{3}) + 17n^{1.2}$
Lời giải

- Ta có:
$a = 10, b = 3$
$f(n) = 17n^{1.2} \in \Theta(n^{1.2}) \Rightarrow d = 1.2$

- Vì $a > b^d (10 > 3^{1.2})$ nên áp dụng case 3 của dạng đơn giản

- Vậy: $T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\log_3 10})$

14. Đề bài
$T(n) = 7T\left(\frac{n}{2}\right) + n^3$
Lời giải

- Ta có: $a = 7, b = 2$
$f(n) = n^3 \in \Theta(n^3) \Rightarrow d = 3$

- Vì $a < b^d (7 < 2^3)$ nên áp dụng case 1 của dạng đơn giản

- Vậy: $T(n) \in \Theta(n^d) = \Theta(n^3)$

15. Đề bài
$T(n) = 4T\left(\frac{n}{2}\right) + logn$
Lời giải

- Ta có:
$a = 4, b = 2, n^{\log_b a} = n^{\log_2 4} = n^2$
$f(n) = \log n = O(n^c) = O(n^{2-\epsilon})$ với c rất nhỏ và $\epsilon > 0$
Nên áp dụng case 1 của dạng tổng quát

- Vậy: $T(n) = \Theta(n^{\log_b a}) = \Theta(n^2)$

16. Đề bài
$T(n) = 4T(\frac{n}{5}) + logn$
Lời giải

- Ta có:
$a = 4, b = 5, n^{\log_b a} = n^{\log_5 4}$
$f(n) = \log n = O(n^c) = O(n^{\log_5 4 - \epsilon})$ với c rất nhỏ và $\epsilon > 0$

- Nên áp dụng case 1 của dạng tổng quát

- Vậy: $T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\log_5 4})$

17. **Đề bài**
$T(n) = \sqrt{2}T(\frac{n}{2}) + \log n$
Lời giải

- Ta có:
$a = \sqrt{2}, b = 2, n^{\log_b a} = n^{\log_2 \sqrt{2}} = n^{\frac{1}{2}}$
$f(n) = \log n = O(n^c) = O(n^{\frac{1}{2}-\epsilon})$ với c rất nhỏ và $\epsilon > 0$
- Nên áp dụng case 1 của dạng tổng quát
- Vậy: $T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\frac{1}{2}})$

18. **Đề bài**
$T(n) = 2T\left(\frac{n}{3}\right) + nlogn$
Lời giải

- Ta có: $a = 2, b = 3, n^{\log_b a} = n^{\log_3 2}$
- Ta thấy: $f(n) = nlogn = \Omega(n^{\log_3 2+\epsilon})$ với $\epsilon > 0$
và $af(n/b) \leq cf(n)$
vì$(2.f(n/3) = 2\frac{n}{3}log\frac{n}{3} = \frac{2}{3}nlogn - \frac{2}{3}n.log3 \leq c.nlogn, c = \frac{2}{3} < 1)$
- Nên áp dụng case 3 của dạng tổng quát
- Vậy: $T(n) = \Theta(f(n)) = \Theta(nlogn)$

19. **Đề bài**
$T(n) = 3T\left(\frac{n}{4}\right) + nlogn$
Lời giải

- Ta có: $a = 3, b = 4, n^{\log_b a} = n^{\log_4 3}$
- Ta thấy: $f(n) = n.logn = \Omega(n^{log_4 3+\epsilon})$ với $\epsilon > 0$
và $af(n/b) \leq cf(n)$
vì $(3.f(n/4) = 3\frac{n}{4}log\frac{n}{4} = \frac{3}{4}nlogn - \frac{3}{2}n \leq c.nlogn, c = \frac{3}{4} < 1)$
- Nên áp dụng case 3 của dạng tổng quát
- Vậy: $T(n) = \Theta(f(n)) = \Theta(nlogn)$

20. **Đề bài**
$T(n) = 6T(\frac{n}{3}) + n^2 logn$
Lời giải

- Ta có: $a = 6, b = 3, n^{log_b a} = n^{log_3 6}$

- Ta thấy: $f(n) = n^2logn = \Omega(n^{log_3 6+\epsilon})$ với $\epsilon > 0$
  và $af(n/b) \leq cf(n)$
  vì $(6.f(n/3) = 6(\frac{n}{3})^2log\frac{n}{3} = \frac{2}{3}n^2(logn - log3) \leq c.n^2logn, c = \frac{2}{3} < 1)$
- Nên áp dụng case 3 của dạng tổng quát
- Vậy: $T(n) = \Theta(f(n)) = \Theta(n^2logn)$

21. Đề bài
$T(n) = 3T(\frac{n}{5}) + \log^2 n$
Lời giải

- Ta có:
  $a = 3, b = 5, n^{\log_b a} = n^{\log_5 3}$
  $f(n) = \log^2 n = O(n^{2c}) = O(n^{\log_5 3-\epsilon})$ với c rất nhỏ và $\epsilon > 0$
- Nên áp dụng case 1 của dạng tổng quát
- Vậy: $T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\log_5 3})$

22. Đề bài
$T(n) = 2T(\frac{n}{2}) + \frac{n}{\log n}$
Lời giải

- Ta có:
  $a = 2, b = 2, \frac{n}{\log n} = n^{\log_b a}\log^{-1} n = n^{\log_2 2}\log^{-1} n = n\log^{-1} n$
  $f(n) = n\log^{-1} n = \Theta(n\log^{-1} n) = \Theta(n^{\log_b a}\log^k n)$
- Không áp dụng được định lý Master dạng đơn giản vì $f(n)$ không phải là đa thức.
- Không áp dụng được định lý Master dạng tổng quát vì ta thấy $k = -1 < 0$

23. Đề bài
$T(n) = 2^nT\left(\frac{n}{2}\right) + n^n$
Lời giải

- Ta có: $a = 2^n$ không phải là hằng số
- Vậy không áp dụng được định lý Master

24. Đề bài
$T(n) = 0.5T(\frac{n}{2}) + n$
Lời giải

- Ta có: $a = 0.5 < 1, b = 2, f(n) = n$

- Vì $a < 1$ nên không áp dụng được định lý Master

25. **Đề bài**
    $T(n) = T(\frac{n}{2}) + n(2 - \cos n)$
    Lời giải

- Ta có:
  $a = 1, b = 2, n^{\log_b a} = n^{\log_2 1} = n^0$
  $f(n) = n(2 - \cos n) = 2n - n\cos n = \Omega(n^{\epsilon})$ với $\epsilon > 0$ (case 3)

- Vì $f(n)$ không phải là hàm đa thức và $f(n)$ có chứa $\cos n$ làm cho $T(n)$ không đơn điệu nên không thể áp dụng định lý Master dạng đơn giản

- Ta có: $af(n/b) \leq cf(n) \Leftrightarrow f(n/2) \leq c.f(n)$
  $\Leftrightarrow n - \frac{n}{2}.\cos\frac{n}{2} \leq c.(2n - n\cos n)$
  $\Leftrightarrow \dfrac{2 - \cos\frac{n}{2}}{2(2 - \cos n)} \leq c$
  Vì $\dfrac{2 - \cos\frac{n}{2}}{2(2 - \cos n)} \leq \dfrac{3}{2}$ nên không tồn tại $c < 1$ với mọi n đủ lớn để $af(n/b) \leq cf(n)$
  Suy ra không thể áp dụng định lý Master dạng tổng quát

26. **Đề bài**
    $T(n) = 64T(\frac{n}{8}) - n^2\log n$
    Lời giải
    Ta có: $f(n) = -n^2\log n$

- Không áp dụng được định lý Master dạng đơn giản vì $f(n)$ không phải là đa thức.

- Không áp dụng được định lý Master dạng tổng quát vì $f(n)$ không phải là 1 hàm tiệm cận dương.

27. **Đề bài**
    $T(n) = T\left(\frac{n}{2}\right) + 2^n$
    Lời giải

- Ta có:
  $a = 1, b = 2, n^{\log_b a} = n^{\log_2 1} = 1$

- Ta thấy: $f(n) = 2^n = \Omega(n^{\epsilon})$ với $\epsilon > 0$
  và $af(n/b) \leq cf(n)$
  vì $(1.f(n/2) = 2^{n/2} \leq c.2^n, c = \dfrac{9}{10} < 1)$
  Nên áp dụng case 3 của dạng tổng quát

- Vậy: $T(n) = \Theta(f(n)) = \Theta(2^n)$

28. Đề bài
$T(n) = 16T(\dfrac{n}{4}) + n!$
Lời giải

- Ta có:
  $a = 16, b = 4, n^{log_4 16} = n^2$
  $f(n) = n! = \Omega(n^{2+\epsilon})$ với $\epsilon > 0$

- Ta có: $af(n/b) \leq cf(n) \Leftrightarrow 16.f(n/4) \leq c.f(n)$
  $\Leftrightarrow 16.(n/4)! \leq c.n!$
  $\Leftrightarrow \dfrac{16.(n/4)!}{n!} \leq c$
  Ta có: $c = \dfrac{2}{3} < 1$ với mọi n đủ lớn để $af(n/b) \leq cf(n)$
  nên áp dụng case 3 của dạng tổng quát

- Vậy: $T(n) = \Theta(f(n)) = \Theta(n!)$