# Discussion 2

https://www.notion.so/Discussion-2-6ed68e7227644709b98a62bdbd2b730b

Haochen Yang,

yhcyang@ucdavis.edu

## FAQ on homework

## Roadmap

1. Quick review
2. Details in textbook
3. Recurrence
4. Divide and Conquer

## Quick Review of the last lecture

- **Why we need asymptotic notation:**

  Compare two algorithm with **time** and **space** complexity

- **Asymptotic notation:**
  - $\Omega$ → lower bound
  - $O$ → Upper bound → Mostly using
  - $\Theta$ → Combination of both

## Some details in the textbook

1. $\omega$ **and** $o$ **(Not commonly used)**

   $o$ → upper bound excluding the exact bound

   $f(n) = 2n^2 \notin o(n^2)$, but $2n^2 \in O(n^2)$

2. **Properties of logarithm and exponential:**

$$\lim_{n\to\infty} \frac{n^b}{a^n} = 0$$

→ Exponential grows faster than polynomial

$$\log_b a = \frac{\log_c a}{\log_c b}, \quad \log_b a = \frac{1}{\log_a b}$$

→ We will use that when proving "why we do not care about the base in logarithm"

3. **Base of exponential:**

> Is $2^{n+1} = O(2^n)$? Is $2^{2n} = O(2^n)$? → Exercise 3.1-4, Page 53

Remember "why base is not important in logarithm":

$$\log_a n = \frac{\log_c n}{\log_c a} = constant \times \log_c n$$

Back to this question:

$$2^{n+1} = 2 \times 2^n = constant \times 2^n \to 2^{n+1} = O(2^n)$$

$$2^{2n} = 2^{2^n} = 4^n \to 4^n \neq O(2^n)$$

4. **How to prove $6n^3 \neq O(n^2)$ ?**

Remember "How to prove $6n^2 = O(n^2)$" ?

→ Find a proper $c$ and $n_0$ that meet the requirements.

Back to the question: (Prove by contradiction)

→ $6n^3 < cn^2$ → $6n < c$ → Not possible

# Recurrence(factorial)

General formula:

```
unsigned int factorial(unsigned int n) {
  if(n == 1)
    return 1;
  return factorial(n-1) * n;
}
```

$$factorial(n) = \begin{cases} 1 & ,n = 1 \\ factorial(n-1) \times n & ,n > 1 \end{cases}$$

$$T(n) = T(n-1) + c = T(n-2) + c + c = \cdots$$
$$T(n) = T(n-(k-1)) + (k-1)c$$

For the last recursion:

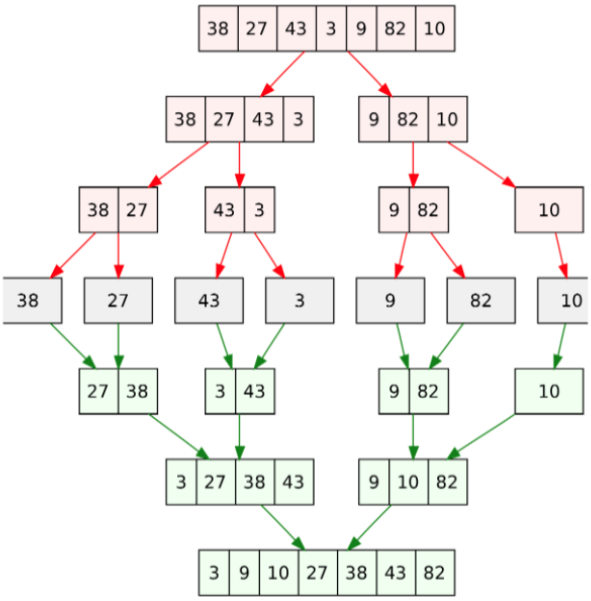
$$T(n) = T(1) + (n-1)c$$
$$T(n) = O(n)$$

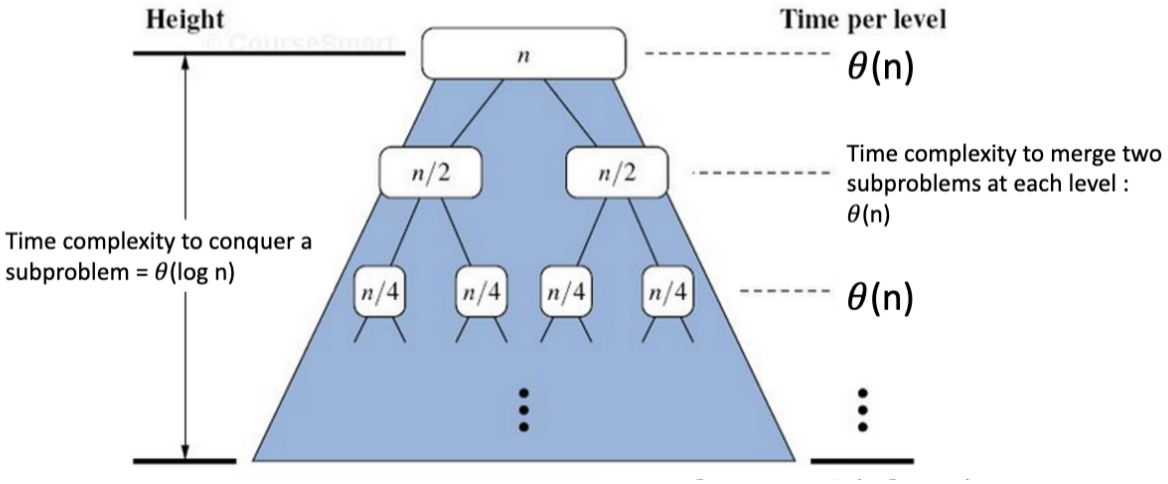
Make a General formula → Mathematical derivation to base case → Simplify the equation, ignore not important items → Get the answer

## Divide and Conquer $(n \log n)$[Optional]

### Divide and Conquer Algorithms

- It refers to a class of algorithmic techniques where the input is partitioned into two or more smaller sub-problems, then it solves each subproblem in a recursive fashion, and then combines the solutions to these subproblems into an overall solution.
- So it has three main steps:
  1. Divide
  2. Conquer
  3. Combine
- Divide and Conquer technique may reduce the running time of some natural brute-force algorithms.
- Example: Merge Sort Algorithm






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

$$T(n) = 2T(\frac{n}{2}) + n = 2(2T(\frac{n}{4}) + \frac{n}{2}) + n = \cdots$$
$$T(n) = 2^k T(\frac{n}{2^k}) + kn$$
$$\frac{n}{2^k} = 1 \rightarrow k = log_2 n$$
$$T(n) = 2^{log_2 n} T(1) + log_2 n = n + n log_2 n = O(n \log n)$$

**Note**: Those mathematic derivations are just help you to understand the analysis

## Tips on hw1:

- Choose all possbile answers
- $2^n < e^n, \log_2 n > log_e n$