Insertion vs. Merge sort

- For big n, merge sort (with $\theta(nlogn)$) beat insertion sort (with $\theta(n^2)$)
- we can sometimes determine the exact running time of an algorithm (as we did for insertion sort)
- the extra precision is not usually worth the effort of computing it
- Neglectable:
  - multiplicative constants
  - lower-order terms
- We use several notations to compare various functions complexity ($o, O, \theta, \Omega, \omega$)

Complexity notations

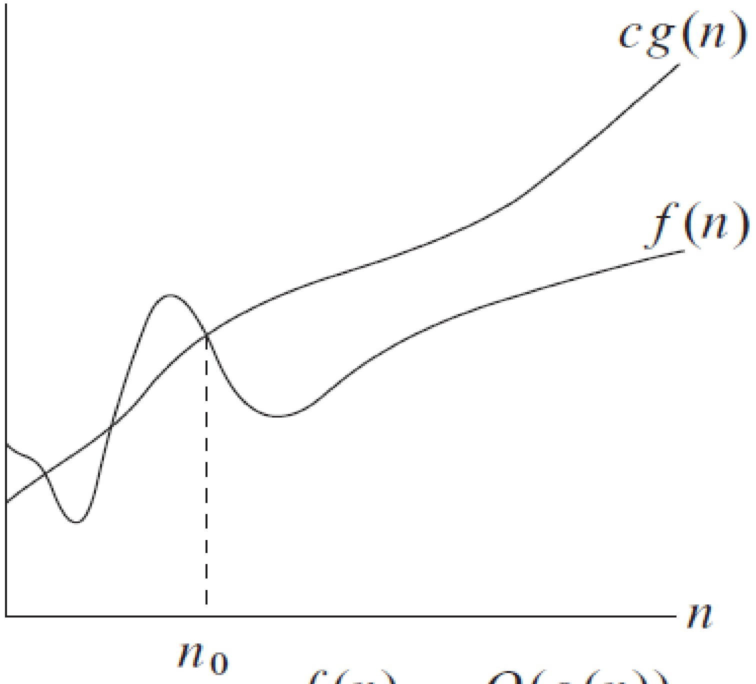
- We use several notations to compare various functions complexity $(o, O, \theta, \Omega, \omega)$
- We use equivalent notations for comparison numbers: $<$ $\leq$ $=$ $\geq$ $>$
- e.g. $a \leq b$ for numbers and $f(n) = O(g(n))$ for functions

$$
\begin{array}{lll}
f(n) = O(g(n)) & \text{is like} & a \leq b \\
f(n) = \Omega(g(n)) & \text{is like} & a \geq b \\
f(n) = \Theta(g(n)) & \text{is like} & a = b \\
f(n) = o(g(n)) & \text{is like} & a < b \\
f(n) = \omega(g(n)) & \text{is like} & a > b
\end{array}
$$

O-notation

$O(g(n)) = \{f(n) :$ there exist positive constants c and n_0 such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0\}$.

$$f(n) = O(g(n))$$



$f(n) = O(g(n))$

O-notation-Example

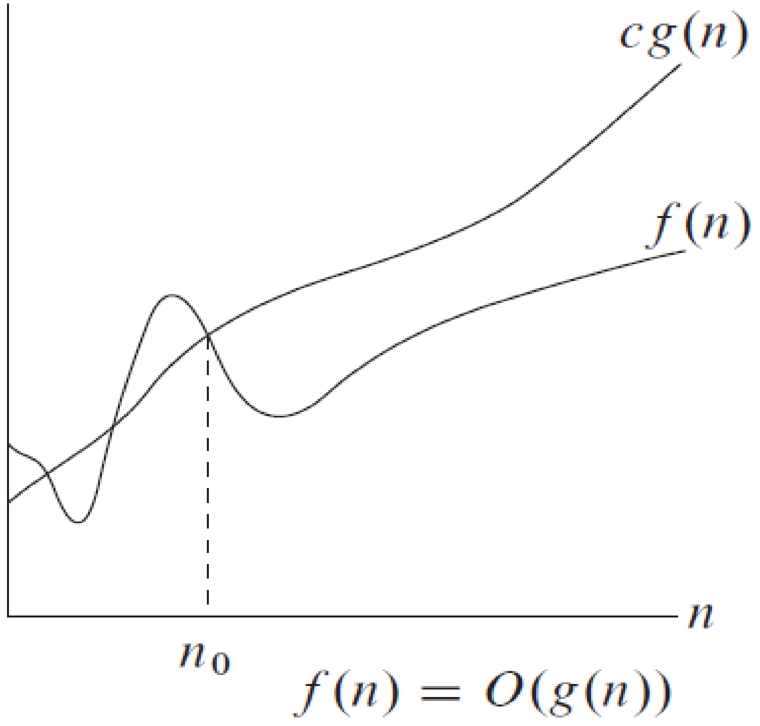
- $f(n) = 2n^2 - 3n + 4$
- $g(n) = n^3$
- $f(n) \stackrel{?}{=} O(g(n))$
- $g(n) \stackrel{?}{=} O(f(n))$

$$2n^2 - 3n + 4 \le cn^3$$

$$\left\{\begin{array}{l} 2n^2 < 2n^3 \\ -3n + 4 < 0 \end{array}\right. \quad n \ge 2$$

$$2n^2 - 3n + 4 \le 2n^3$$

$$n_0 = 2$$
$$C = 2$$



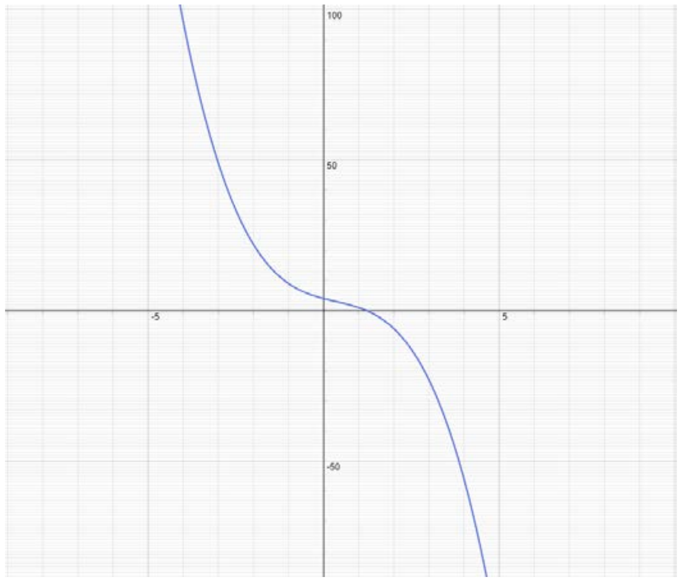
O-notation-Example

- $\mathrm{g}(n) \stackrel{?}{=} O(f(n))$

$\nexists n_0, c > 0 \qquad n > n_0 \qquad n^3 < c(n^2 - 3n + 4)$

$0 < -n^3 + cn^2 - 3cn + 4c$

$n = n_0 + c$



O-notation-Example

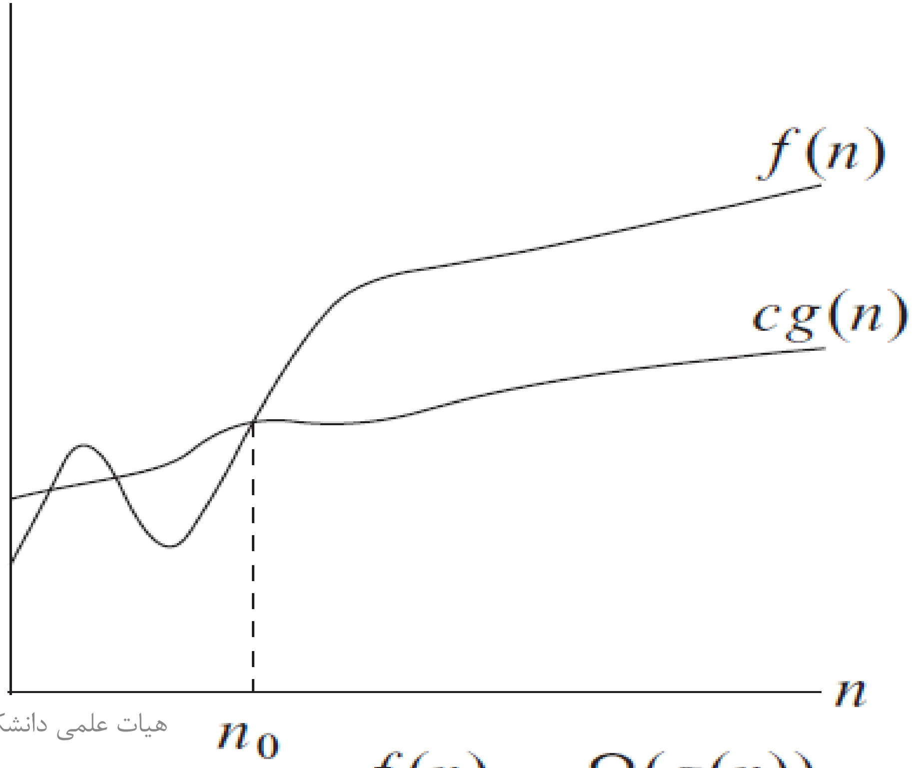
- $f(n) = 2^{10^9} n^2$
- $g(n) = n^2$
- $g(n) \stackrel{?}{=} O(f(n))$
- $f(n) \stackrel{?}{=} O(g(n))$

Ω-notation

$$\Omega(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \le cg(n) \le f(n) \text{ for all } n \ge n_0\}\ .$$

$$\boldsymbol{f(n) = \Omega(g(n))}$$

- $f(n) = 2n^2 - 3n + 4$
- $g(n) = n^3$
- $f(n) = O(g(n))$
- $g(n) = \Omega(f(n))$



$f(n) = \Omega(g(n))$

سمانه حسینی سمنانی
هیات علمی دانشکده برق و کامپیوتر- دانشگاه صنعتی اصفهان

Ω-notation-Example

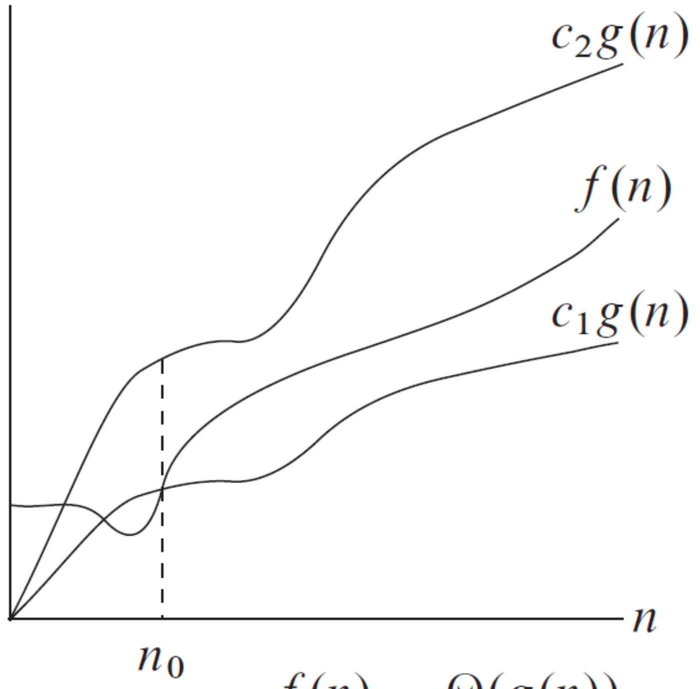
- $f(n) = 2^{10^9} n^2$
- $g(n) = n^2$
- $g(n) \stackrel{?}{=} \Omega(f(n))$
- $f(n) \stackrel{?}{=} \Omega(g(n))$

θ-notation

$$\Theta(g(n)) = \{f(n) : \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \text{ such that } 0 \le c_1 g(n) \le f(n) \le c_2 g(n) \text{ for all } n \ge n_0\}\ .^1$$

A function $f(n)$ belongs to the set $g(n)$ if there exist positive constants c_1 and c_2 such that it can be be "sandwiched" between $c_1 g(n)$ and $c_2 g(n)$, for sufficiently large n.

$$f(n) = \Theta(g(n))$$



$f(n) = \Theta(g(n))$

θ-notation-Example

- $f(n) = 2^{10^9} n^2$
- $g(n) = n^2$
- $g(n) \stackrel{?}{=} \theta\,(f(n))$
- $f(n) \stackrel{?}{=} \theta(g(n))$

Theorem 3.1

For any two functions $f(n)$ and $g(n)$, we have $f(n) = \Theta(g(n))$ if and only if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$. ■

- $an^2 + bn + c = \theta(n^2)$
- $an^2 + bn + c = O(n^2)$
- $an^2 + bn + c = \Omega(n^2)$

Theorem

$f(n) = a_k n^k + \cdots + a_1 n + a_0 \quad \rightarrow \quad f(n) = \theta\left(n^k\right) \quad a_k > 0$

$f(n) = O(n^k) \quad \rightarrow \exists c, n_0 > 0 \quad a_k n^k + \cdots + a_1 n + a_0 \leq c n^k$

$c = \sum_{i=0}^{k} |a_i| \quad , \quad n_0 = 1$

$f(n) = \Omega(n^k) \quad \rightarrow \quad \exists c, n_0 > 0 \quad a_k n^k + \cdots + a_1 n + a_0 \geq c n^k$

o-notation

$$O(g(n)) = \{f(n) : \text{ there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \le f(n) \le cg(n) \text{ for all } n \ge n_0\}.$$

$$o(g(n)) = \{f(n) : \text{ for any positive constant } c > 0, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \le f(n) < cg(n) \text{ for all } n \ge n_0\}.$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

For example, $2n = o(n^2)$, but $2n^2 \ne o(n^2)$

o-notation-Example

$$o(g(n)) = \{f(n) : \text{ for any positive constant } c > 0, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \le f(n) < cg(n) \text{ for all } n \ge n_0\}.$$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

- $f(n) = 2n^2$
- $g(n) = 3n^2 + 5n + 6$
- $f(n) \neq o(g(n))$

برهان خلف: $\forall\, c \;\; \exists n_0 \quad n > n_0 \qquad 2n^2 < c(3n^2 + 5n + 6)$

$$C = 1/3$$

ω-notation

$$\Omega(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \le cg(n) \le f(n) \text{ for all } n \ge n_0\} .$$

$$\omega(g(n)) = \{f(n) : \text{for any positive constant } c > 0, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \le cg(n) < f(n) \text{ for all } n \ge n_0\} .$$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty$$

For example, $n^2/2 = \omega(n)$, but $n^2/2 \ne \omega(n^2)$

Comparing functions

- Many of the relational properties of real numbers apply to asymptotic comparisons as well:
  - Transitivity: تعدی
  - Reflexivity: انعکاسی
  - Symmetry: تقارنی
  - Transpose symmetry

Transitivity یا تعدی

$$f(n) = \Theta(g(n)) \text{ and } g(n) = \Theta(h(n)) \quad \text{imply} \quad f(n) = \Theta(h(n))$$

$$f(n) = O(g(n)) \text{ and } g(n) = O(h(n)) \quad \text{imply} \quad f(n) = O(h(n))$$

$$f(n) = \Omega(g(n)) \text{ and } g(n) = \Omega(h(n)) \quad \text{imply} \quad f(n) = \Omega(h(n))$$

$$f(n) = o(g(n)) \text{ and } g(n) = o(h(n)) \quad \text{imply} \quad f(n) = o(h(n))$$

$$f(n) = \omega(g(n)) \text{ and } g(n) = \omega(h(n)) \quad \text{imply} \quad f(n) = \omega(h(n))$$

Reflexivity یا انعکاسی

$$
\begin{aligned}
f(n) &= \Theta(f(n)) \\
f(n) &= O(f(n)) \\
f(n) &= \Omega(f(n))
\end{aligned}
$$

سمانه حسینی سمنانی
هیات علمی دانشکده برق و کامپیوتر- دانشگاه صنعتی اصفهان

Symmetry یا تقارن

$$f(n) = \Theta(g(n)) \text{ if and only if } g(n) = \Theta(f(n))$$

Transpose symmetry

$$f(n) = O(g(n)) \text{ if and only if } g(n) = \Omega(f(n))$$

$$f(n) = o(g(n)) \text{ if and only if } g(n) = \omega(f(n))$$

Complexity comparison

$$f(n) = \begin{cases} 2^n & n \le 10^{10} \\ 100n & n > 10^{10} \end{cases}$$

$$g(n) = 2n^2 + n$$

- $\mathrm{g}(n) \stackrel{?}{=} O(f(n))$
- $f(n) \stackrel{?}{=} O(g(n))$

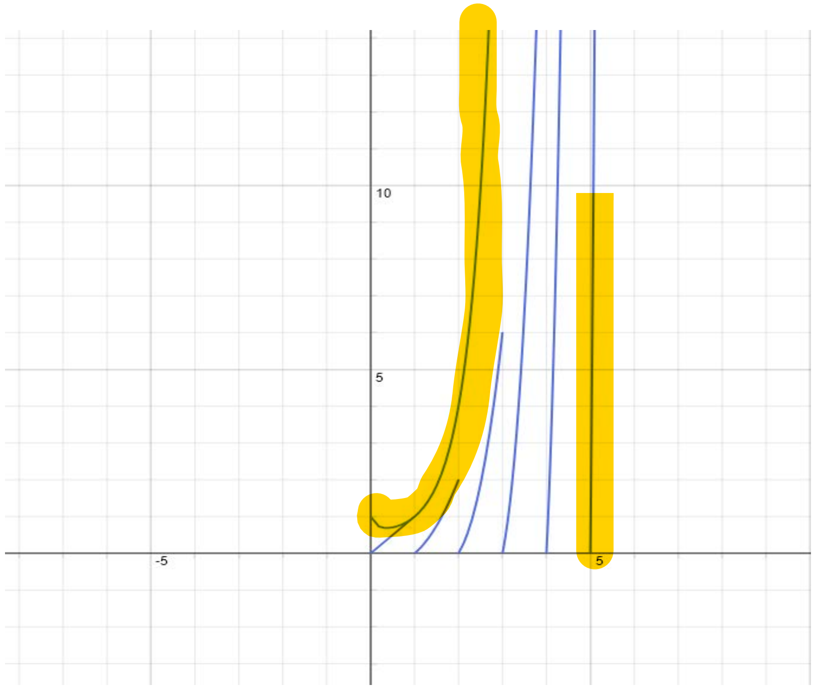
Complexity comparison

$n! \quad = o \quad (n^n)$

$n! \quad (2^n)$

$3^n \quad (2^n)$

$\log n! \quad (nlogn)$



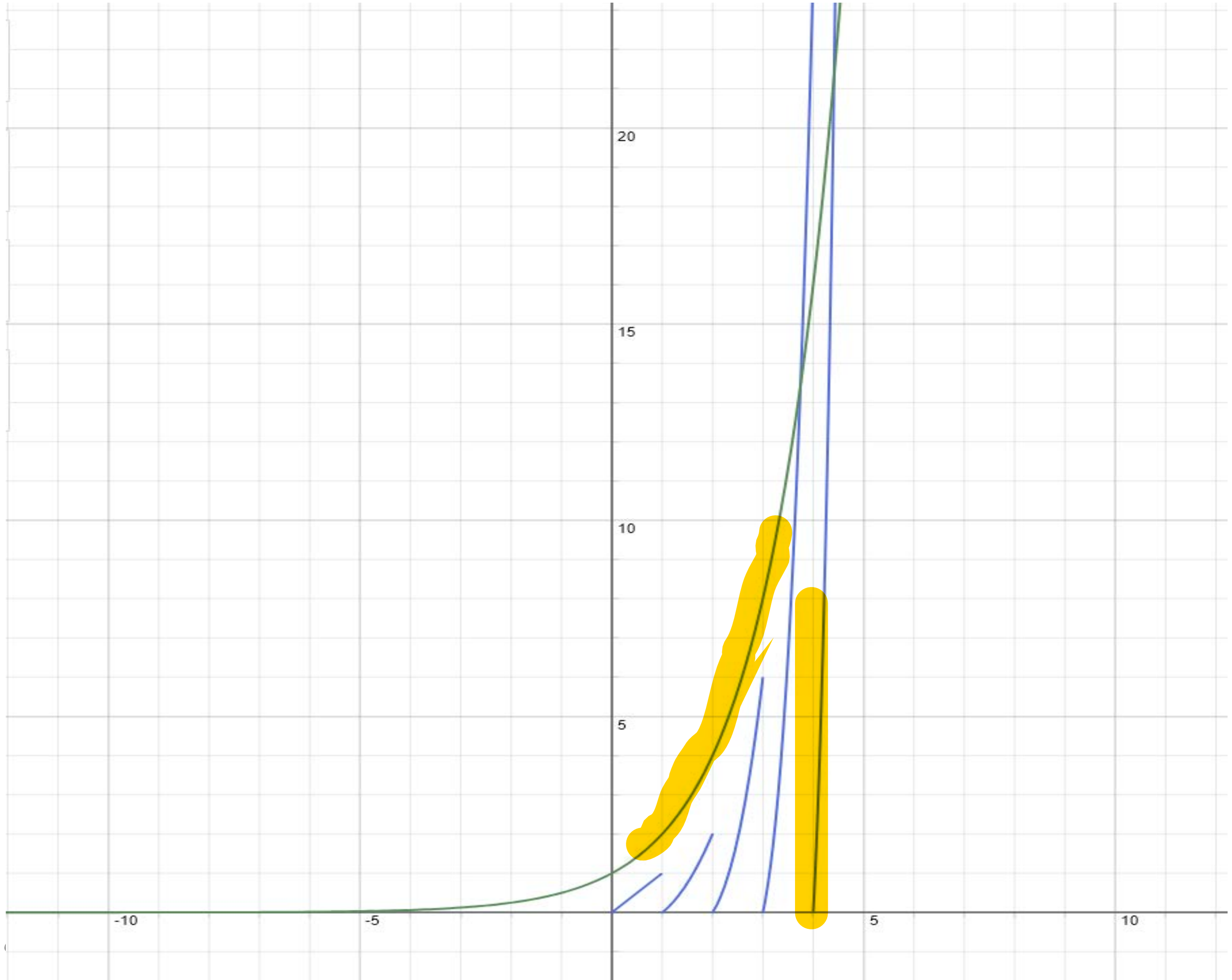
Complexity comparison

$n! \quad = o \quad (n^n)$

$n! \quad = \omega \quad (2^n)$

$3^n \quad\quad (2^n)$

$\log n! \quad\quad (nlogn)$



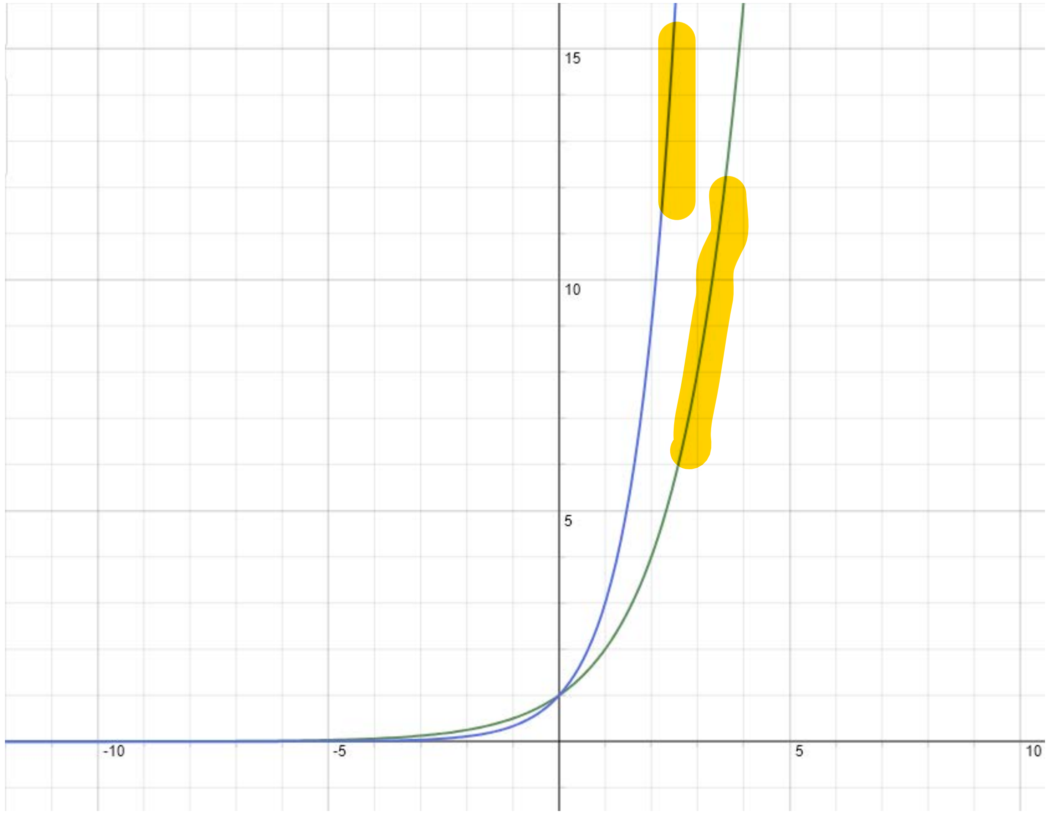
Complexity comparison

$n! \quad = o \quad (n^n)$

$n! \quad = \omega \quad (2^n)$

$3^n \quad = \omega \quad (2^n)$

$\log n! \quad\quad (nlogn)$



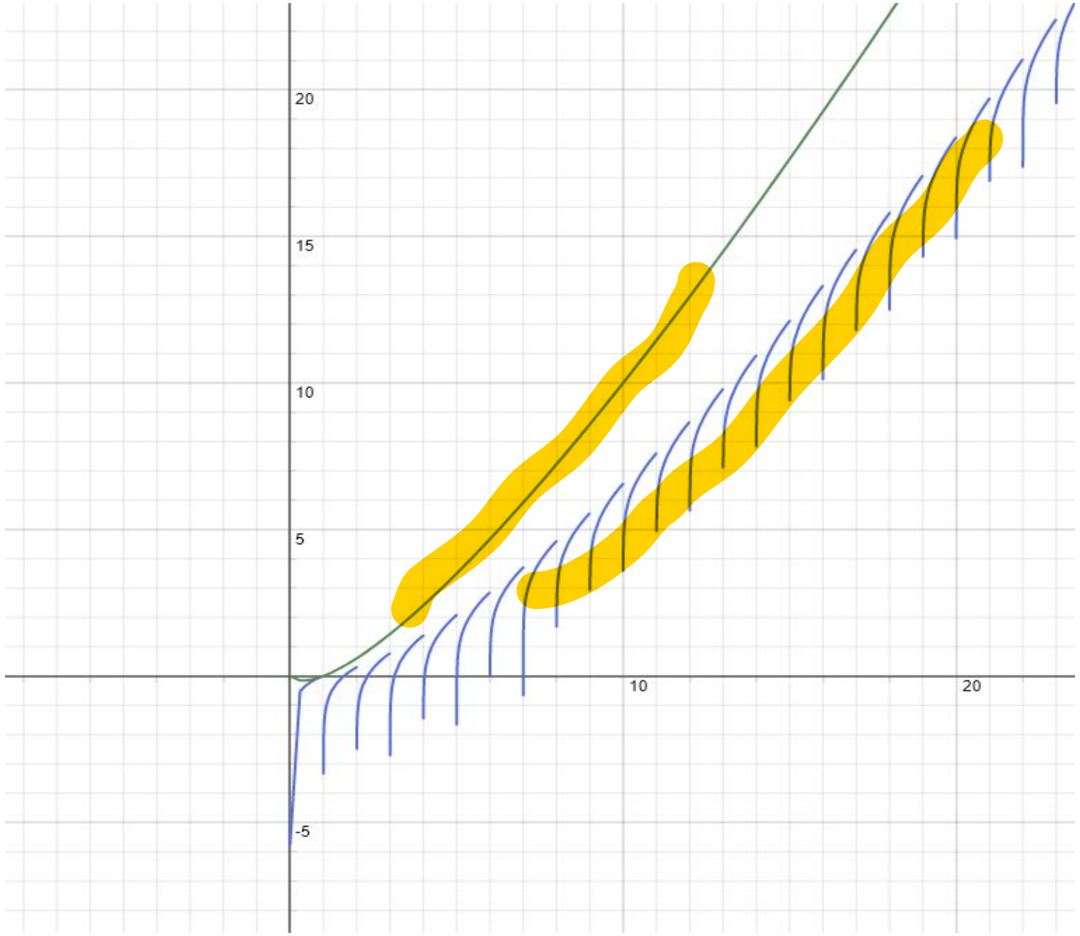
Complexity comparison

$n! \quad = o \quad (n^n)$

$n! \quad = \omega \quad (2^n)$

$3^n \quad = \omega \quad (2^n)$

$\log n! \quad = \theta \quad (n log n)$



Complexity comparison

- Which one has the lowest complexity?

n^2 $\quad n!$ $\quad \log n\,!$ $\quad 2^{2^n}$ $\quad (3/2)^n$

n^3 $\quad n^{\frac{1}{\log n}}$ $\quad n^{0.0001}$ $\quad \log n^{1000}$

Complexity comparison

$$f(n) = O(g(n)) \xrightarrow{?} 2^{f(n)} = O\left(2^{g(n)}\right)$$

$$f(n) = 2n$$

$$g(n) = n$$

$$f(n) = O(g(n)) \xrightarrow{?} \log f(n) = O(\log(g(n)))$$