CENG 218
Design and Analysis of Algorithms

Izmir Institute of Technology

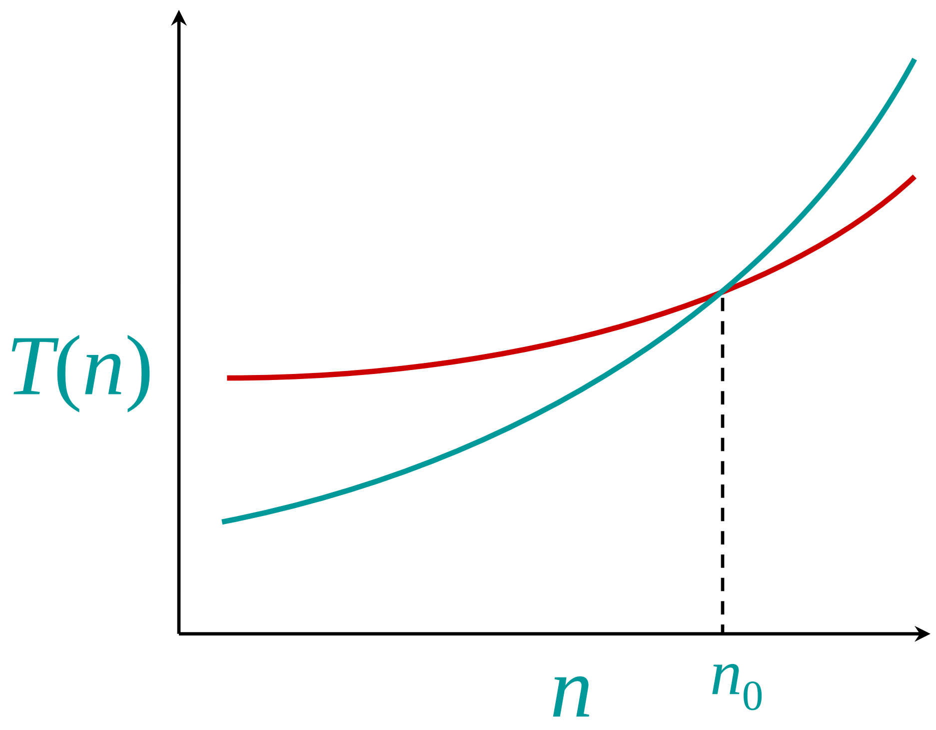
Lecture 2: Asymptotic Notation

Growth of functions

- Previously, we have seen to compute the running time of an algorithm in terms of input size, *n*.
- The order of growth of that time, $T(n)$, gives an idea about the algorithm's efficiency.
- Thus, we can compare the *relative* performance of alternative algorithms.
- Useful in engineering for showing that one design *scales* better or worse than another.

Growth of functions

When n gets large enough, a $\Theta(n^2)$ algorithm *always* beats a $\Theta(n^3)$ algorithm.

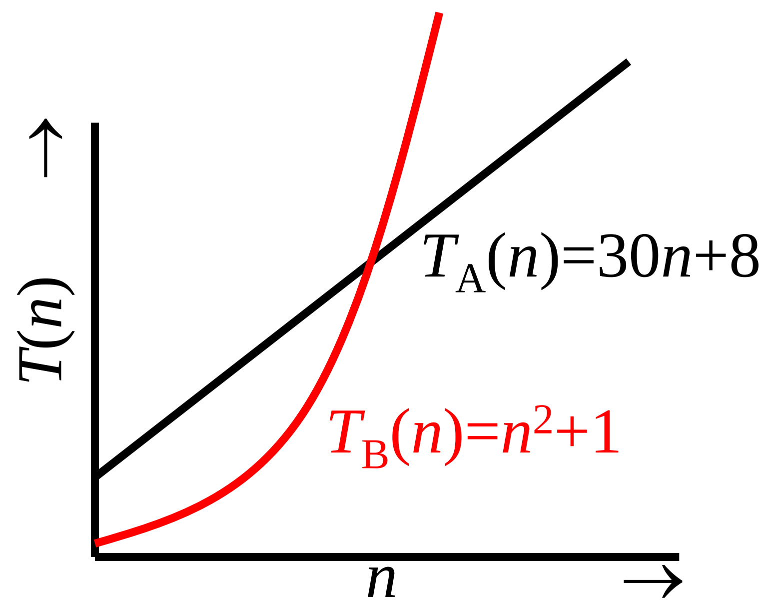


Asymptotic performance

- To make a comparison, we look at input sizes that are large enough.
- That is, the size of the input *in the limit*.
  I.e. The growth of $T(n)$ as $n \to \infty$.
- This is why we call it 'asymptotic' performance.

Asymptotic performance: Example

- Suppose you are designing a web site to process database records of n users.
- Program A takes $T_A(n)=30n+8$ microseconds, while program B takes $T_B(n)=n^2+1$.
- Which program do you choose?



Asymptotic analysis

- Asymptotic analysis is a useful tool to structure our thinking toward better algorithms.
- It is independent of the computer/platform.
- We shouldn’t ignore asymptotically slower algorithms, however. They may be faster for reasonable size input.
- Real-world design situations often call for a careful balancing.

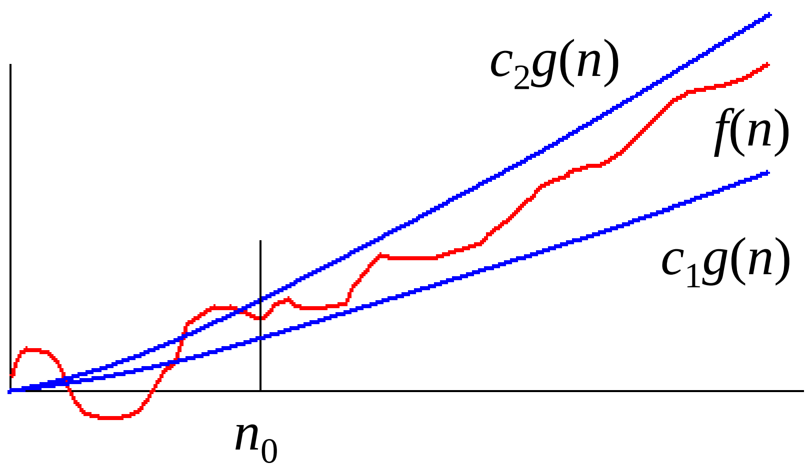
Θ-notation

Mathematical definition:

$f(n) = \Theta(g(n))$ means (read = as 'is')

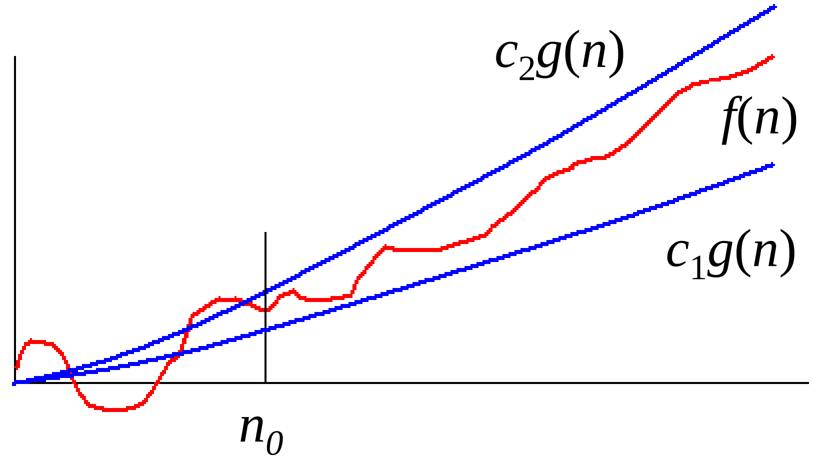
there exist positive constants c_1, c_2, and n_0 such that

$c_1\, g(n) \le f(n) \le c_2\, g(n)$ for all $n \ge n_0$



$f(n)$ is 'sandwiched' between $c_1\, g(n)$ and $c_2\, g(n)$ for big enough n

Θ-notation



Example:

$f(n)=8n^3+5n^2+7$. $g(n)=n^3$. What may be c_1, c_2, n_0 ?

- c_1 can be 1 as long as n_0 is positive.
- c_2 =10 and n_0 =3 is a proper pair.
- Therefore, using $(c_1, c_2, n_0)=(1,10,3)$
  we can say that $8n^3+5n^2+7= \Theta(n^3)$

Θ-notation

- If the definition is satisfied, we say that $g(n)$ is an **asymptotically tight bound** for $f(n)$.
- **Important:** The c_1, c_2, n_0 values that make the statement true are *not* unique. Any triple that satisfies the inequalities also work.

Facts about Θ-notation

- When a polynomial function is used,
  the leading (n^{th}) term dominates its growth:
  $f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$,
  then $f(x) = \Theta(x^n)$.

More Θ-notation facts

- Transitivity:
  $f = \Theta(g)$ and $g = \Theta(h) \rightarrow f = \Theta(h)$
- Reflexivity:
  $f = \Theta(f)$
- Symmetry:
  $f = \Theta(g) \leftrightarrow g = \Theta(f)$

More Θ-notation facts

- Sums of functions:
  If $g = \Theta(f)$ and $h = \Theta(f)$, then $g+h = \Theta(f)$.

- Operations with a constant:
  $\forall a>0,\ \Theta(af) = \Theta(f+a) = \Theta(f-a) = \Theta(f)$

More Θ-notation facts

- Combination of functions:

  f_1 is $\Theta(g_1)$ and f_2 is $\Theta(g_2)$ then,

  1) $f_1 f_2$ is $\Theta(g_1 g_2)$
  2) f_1+f_2 is $\Theta(g_1+g_2) = \Theta(\max(g_1,g_2))$

Θ-notation exercises

1) Give the Θ estimate for $(n\log n + n^2)(n^3 + 2)$

Solution: $(n\log n + n^2)(n^3 + 2) =$

$$(\Theta(n)\cdot\Theta(\log n) + \Theta(n^2))\cdot\Theta(n^3) =$$

$$(\Theta(n\log n) + \Theta(n^2))\cdot\Theta(n^3) =$$

$$\Theta(n^2)\cdot\Theta(n^3) =$$

$$\Theta(n^5)$$

2) Give the Θ estimate for $3n^4 + \log_2 n^8$

Solution: $3n^4 + 8\log_2 n = \Theta(n^4)$

O-notation (Big-O)

- We say $f(n) = \mathrm{O}(g(n))$ if there exist positive constants c and n_0 such that $0 \le f(n) \le c\,g(n)$ for all $n \ge n_0$
  - When n is greater than n_0, function f is at most a constant c times function g
- We say that $g(n)$ is an **asymptotic upper bound** for $f(n)$.
- Following descriptions are also used: "*f* is *at most order g*", or "*f* is big-O *g*"

Big-O definition exercise

- Show that $f(n) = 2n^2+5n+9$ is $O(n^2)$ by finding a pair of (c,n_0) for the definition of the big-O.

$g(n) = n^2$ $\qquad f(n) = 2n^2+5n+9$

For $c=2$, $2n^2+5n+9 \le 2n^2$ is never true.

Let $c=3$, then $2n^2+5n+9 \le 3n^2$ is true when $5n+9 \le n^2$ which corresponds to $n \ge 7$.

Therefore, $(c,n_0)=(3,7)$ is a proper pair (not the only one) to show that $f(n)$ is $O(n^2)$.

Some facts about big-O

- Unlike Θ, $g(n)$ can be higher order than $f(n)$.
- Because big-O denotes upper bound, we can write $20n^2+4 = O(n^5)$.
- It has transitivity:
  $f = O(g)$ and $g = O(h) \rightarrow f = O(h)$
- It has reflexivity:
  $f = O(f)$
- It does NOT have symmetry:
  $f = O(g)$ does not imply $g = O(f)$

Ω-notation (Big omega)

- We say $f(n) = \Omega(g(n))$ if there exist positive constants c and n_0 such that $0 \le c\,g(n) \le f(n)$ for all $n \ge n_0$
  - When n is greater than n_0, function f is at least a constant c times function g
- We say that $g(n)$ is an **asymptotic lower bound** for $f(n)$.
- Following descriptions are also used: "*f* is *at least order g*", or "*f* is big-Ω *g*"

Some facts about Ω

- Since Ω denotes lower bound,
  $g(n)$ can be lower order than $f(n)$.
  E.g. $20n^2+4 = \Omega(n)$
  E.g. $\sqrt{n} = \Omega(\log_2 n)$. Is it?

- Like big-O, Ω has transitivity and reflexivity.
- Like big-O, Ω does NOT have symmetry.

Difference between O and Θ estimates

Question: Give the order of growth for the sum of the first *n* positive integers that are divisible by 3.

Solution 1:

$f(n) = 3+6+9+\dots.+3n \quad < \quad 3n + 3n + 3n + \dots.+3n$

$3+6+9+\dots.+3n \quad < \quad n\cdot(3n)$

$3+6+9+\dots.+3n \quad < \quad 3n^2$

$f(n) = O(n^2)$

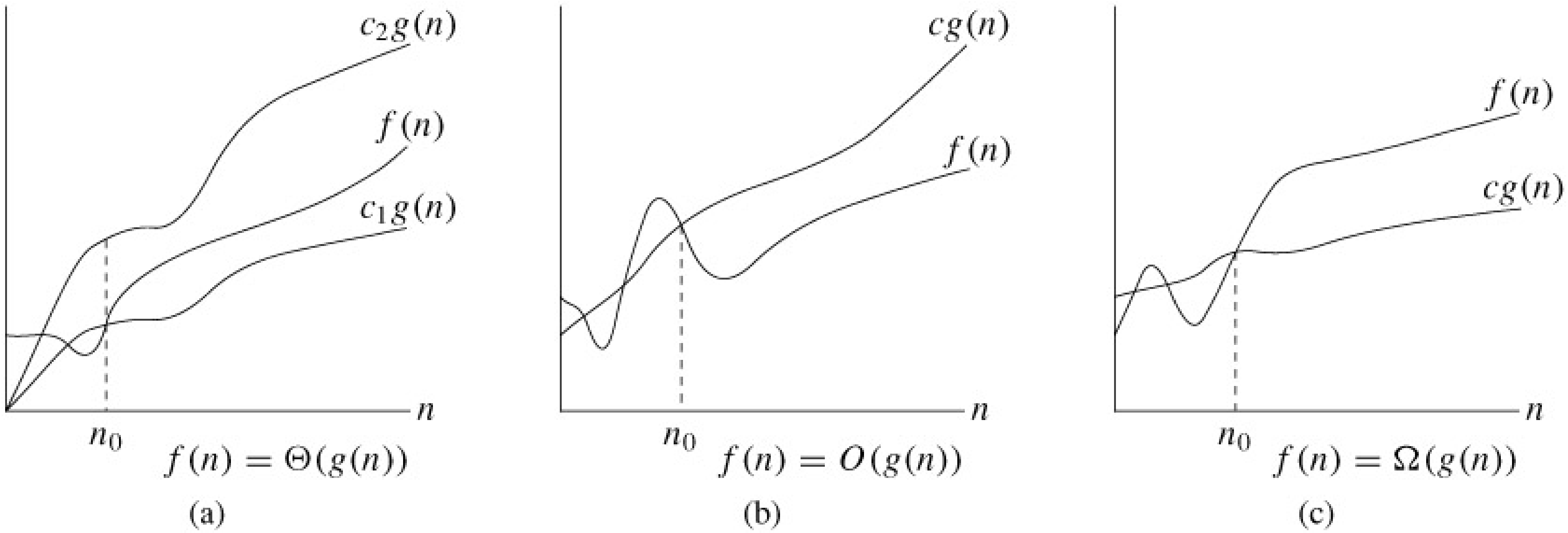
With this solution you can not say $f(n) = \Theta(n^2)$.

To find Θ estimate, we need a closed-form formula.

Solution 2:

$$3\sum_{k=1}^{n} k = 3\frac{(n+1)\cdot n}{2} = (3/2)\cdot(n^2+n) = \Theta(n^2)$$

$\Theta(g)$, $O(g)$, and $\Omega(g)$



$$f(n) = \Theta(g(n)) \Leftrightarrow (f(n) = O(g(n))) \wedge (f(n) = \Omega(g(n)))$$

Macro convention

- "O(f)" when used as a term in an arithmetic expression means: "some function f such that f=O(f)".
- *E.g.*: "x^2+O(x)" means "x^2 plus some function that is O(x)".

o-notation (little-o)

- We use o-notation to denote that the asymptotic upper bound provided by big-O is NOT asymptotically tight.
- E.g.: $2n = o(n^2)$, but $2n^2 \neq o(n^2)$
- $f(n) = o(g(n))$ if for any positive constant $c>0$, there exists n_0 such that $0 \leq f(n) < cg(n)$ for $n \geq n_0$
- Another view:

$$f(n) = o(g(n)) \text{ means } \quad \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

ω-notation (little omega)

- We use ω-notation to denote that the asymptotic lower bound provided by Ω is NOT asymptotically tight.
- E.g.: $n^2/2 = \omega(n)$, but $n^2/2 \neq \omega(n^2)$
- $f(n) = \omega(g(n))$ if for any positive constant $c>0$, there exists n_0 such that $0 \leq cg(n) < f(n)$ for $n \geq n_0$
- Another view:

$$f(n) = \omega(g(n)) \text{ means } \quad \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

More facts

- Analogy with the comparison of two numbers:

  $f(n) = O(g(n))$ is like $a \le b$

  $f(n) = \Omega(g(n))$ is like $a \ge b$

  $f(n) = \Theta(g(n))$ is like $a = b$

  $f(n) = o(g(n))$ is like $a < b$

  $f(n) = \omega(g(n))$ is like $a > b$

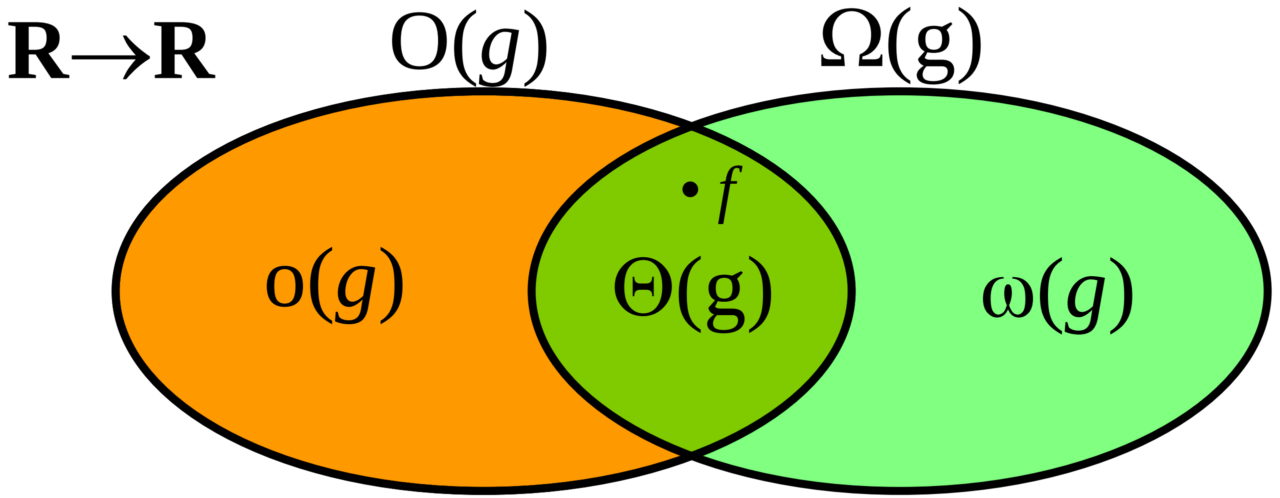
- Transpose symmetry:

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

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

Relations between $\Theta(g)$, $O(g)$, $\Omega(g)$

- Subset relations between order-of-growth sets.



- Big-omega and big-theta notations are introduced by Donald Knuth (1938-...)

Exercises

1) $2^{n+1} = O(2^n)$ T/F?

   True

2) $f(n) + O(f(n)) = \Theta(f(n))$ T/F?

   True

3) $2^n = \omega(2^{n-2})$ T/F?

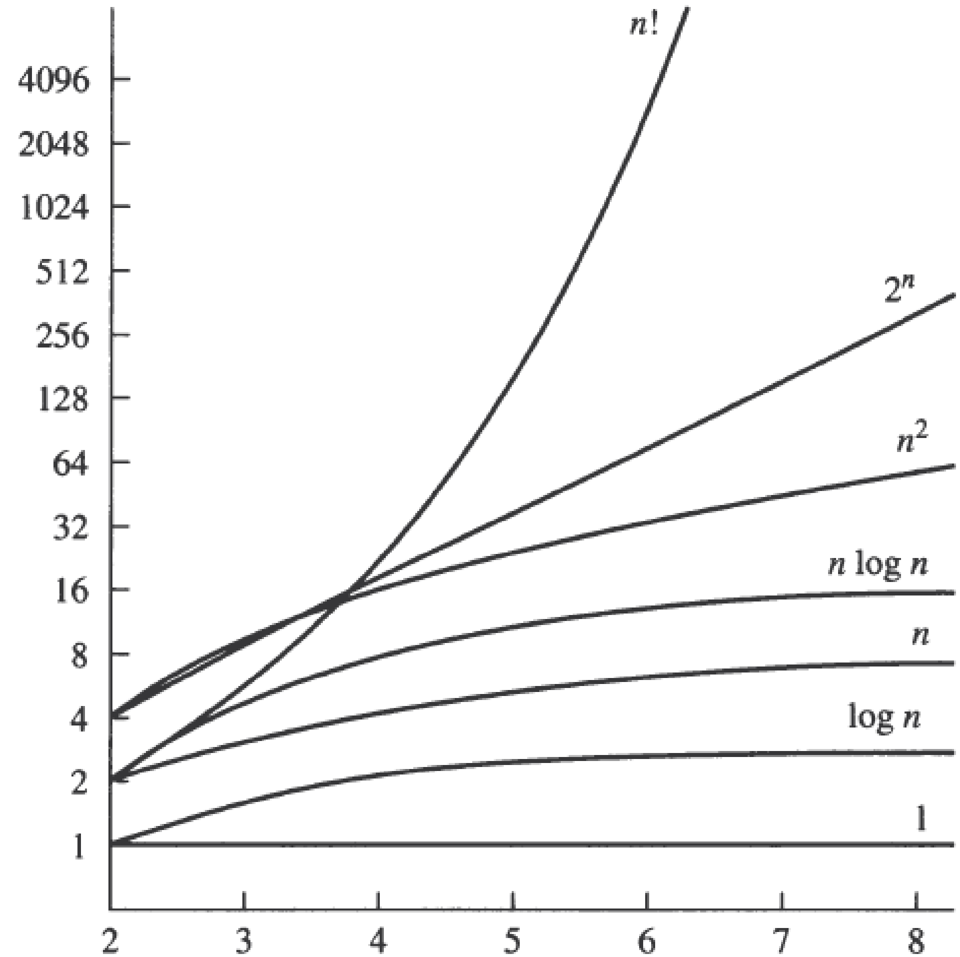
   False

Terminology for the Growth of Functions

• $\Theta(1)$	Constant
• $\Theta(\log_c n)$	Logarithmic (same order $\forall c$)
• $\Theta(\log^c n)$	Polylogarithmic
• $\Theta(n)$	Linear
• $\Theta(n \log n)$	$n \log n$
• $\Theta(n^c)$	Polynomial
• $\Theta(c^n)$, $c>1$	Exponential
• $\Theta(n!)$	Factorial

Ordering of Functions

Let's write $f < g$ for
g is higher order than f
and let k>1 then:
$1 < \log n < n < n \log n$
$< n^k < k^n < n! < n^n$



Computer time examples

#opers	n=10	$n=10^6$
log n	$3\cdot10^{-9}$ s	$2\cdot10^{-8}$ s
n	10^{-8} s	10^{-3} s
n log n	$3\cdot10^{-8}$ s	$2\cdot10^{-2}$ s
n^2	10^{-7} s	17 min
2^n	10^{-6} s	$>10^{300,000}$ years
n!	$3\cdot10^{-3}$ s	****

Assume:
10^{-9} second
per operation.

Quiz

Which ones are more appealing?

$$\underbrace{n^{1000} \qquad n^{2}}_{\text{Polynomials}} \qquad 2^{n}$$

Which ones are more appealing?

$$\underbrace{1000n^{2} \qquad 3n^{2}}_{\text{Quadratic}} \qquad 2n^{3}$$

The End

- Next, we will continue with Divide-and-Conquer approach to analyze recursive algorithms.
- Please solve exercises of Chapter 3.1 and problems of Chapter 3.