Analysis of algorithms

- **Issues:**
  - correctness
  - time efficiency
  - space efficiency
  - optimality

- **Approaches:**
  - theoretical analysis
  - empirical analysis

Theoretical analysis of time efficiency

Time efficiency is analyzed by determining the number of repetitions of the *basic operation* as a function of *input size*

- ***Basic operation*: the operation that contributes most towards the running time of the algorithm**

input size

$$T(n) \approx c_{op}C(n)$$

running time

execution time for basic operation

Number of times basic operation is executed

Input size and basic operation examples

Problem	*Input size measure*	*Basic operation*
Searching for key in a list of *n* items	Number of list's items, i.e. *n*	Key comparison
Multiplication of two matrices	Matrix dimensions or total number of elements	Multiplication of two numbers
Checking primality of a given integer *n*	*n*'size = number of digits (in binary representation)	Division
Typical graph problem	#vertices and/or edges	Visiting a vertex or traversing an edge

Empirical analysis of time efficiency

- Select a specific (typical) sample of inputs

- Use physical unit of time (e.g., milliseconds)

  or

  Count actual number of basic operation's executions

- Analyze the empirical data

Best-case, average-case, worst-case

For some algorithms efficiency depends on form of input:

- **Worst case:** $C_{worst}(n)$ – maximum over inputs of size n
- **Best case:** $C_{best}(n)$ – minimum over inputs of size n
- **Average case:** $C_{avg}(n)$ – "average" over inputs of size n
  - Number of times the basic operation will be executed on typical input
  - NOT the average of worst and best case
  - Expected number of basic operations considered as a random variable under some assumption about the probability distribution of all possible inputs

Example: Sequential search

ALGORITHM *SequentialSearch*$(A[0..n-1], K)$

//Searches for a given value in a given array by sequential search
//Input: An array $A[0..n-1]$ and a search key K
//Output: The index of the first element of A that matches K
// or -1 if there are no matching elements

$i \leftarrow 0$
while $i < n$ **and** $A[i] \neq K$ **do**
    $i \leftarrow i + 1$
if $i < n$ **return** i
else return -1

- Worst case

- Best case

- Average case

Types of formulas for basic operation's count

- **Exact formula**

  e.g., $C(n) = n(n-1)/2$

- **Formula indicating order of growth with specific multiplicative constant**

  e.g., $C(n) \approx 0.5\, n^2$

- **Formula indicating order of growth with unknown multiplicative constant**

  e.g., $C(n) \approx cn^2$

Order of growth

- Most important: Order of growth within a constant multiple as $n \to \infty$

- Example:
  - How much faster will algorithm run on computer that is twice as fast?
  - How much longer does it take to solve problem of double input size?

Values of some important functions as $n \to \infty$

n	$\log_2 n$	n	$n \log_2 n$	n^2	n^3	2^n	$n!$
10	3.3	10^1	$3.3 \cdot 10^1$	10^2	10^3	10^3	$3.6 \cdot 10^6$
10^2	6.6	10^2	$6.6 \cdot 10^2$	10^4	10^6	$1.3 \cdot 10^{30}$	$9.3 \cdot 10^{157}$
10^3	10	10^3	$1.0 \cdot 10^4$	10^6	10^9		
10^4	13	10^4	$1.3 \cdot 10^5$	10^8	10^{12}		
10^5	17	10^5	$1.7 \cdot 10^6$	10^{10}	10^{15}		
10^6	20	10^6	$2.0 \cdot 10^7$	10^{12}	10^{18}		

Table 2.1 Values (some approximate) of several functions important for analysis of algorithms

Asymptotic order of growth

A way of comparing functions that ignores constant factors and small input sizes

- O(g(n)): class of functions f(n) that grow no faster than g(n)
- Θ(g(n)): class of functions f(n) that grow at same rate as g(n)
- Ω(g(n)): class of functions f(n) that grow at least as fast as g(n)

Big-oh

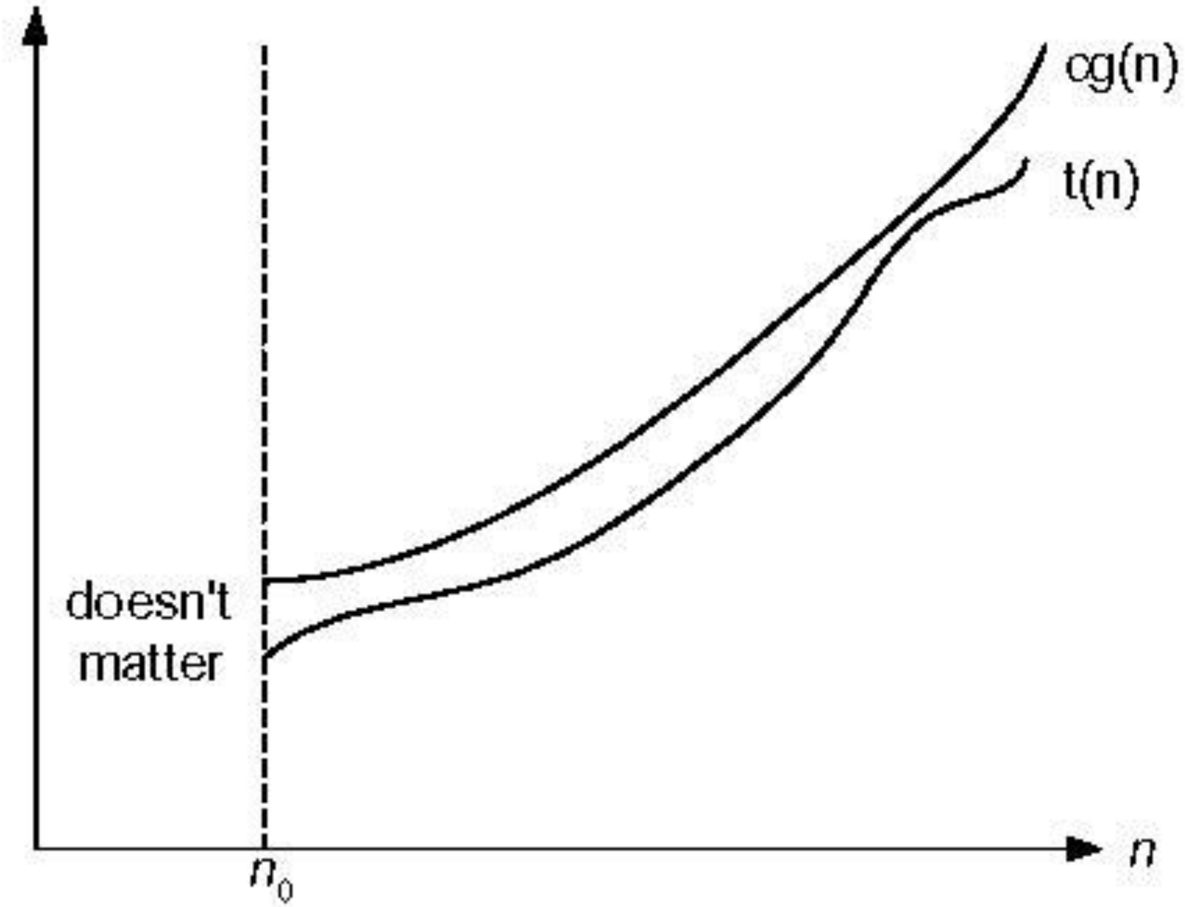


Figure 2.1 Big-oh notation: $t(n) \in O(g(n))$

O-notation

DEFINITION A function $t(n)$ is said to be in $O(g(n))$, denoted $t(n) \in O(g(n))$, if $t(n)$ is bounded above by some constant multiple of $g(n)$ for all large n, i.e., if there exist some positive constant c and some nonnegative integer n_0 such that

$$t(n) \leq cg(n) \quad \text{for all } n \geq n_0.$$

Big-omega

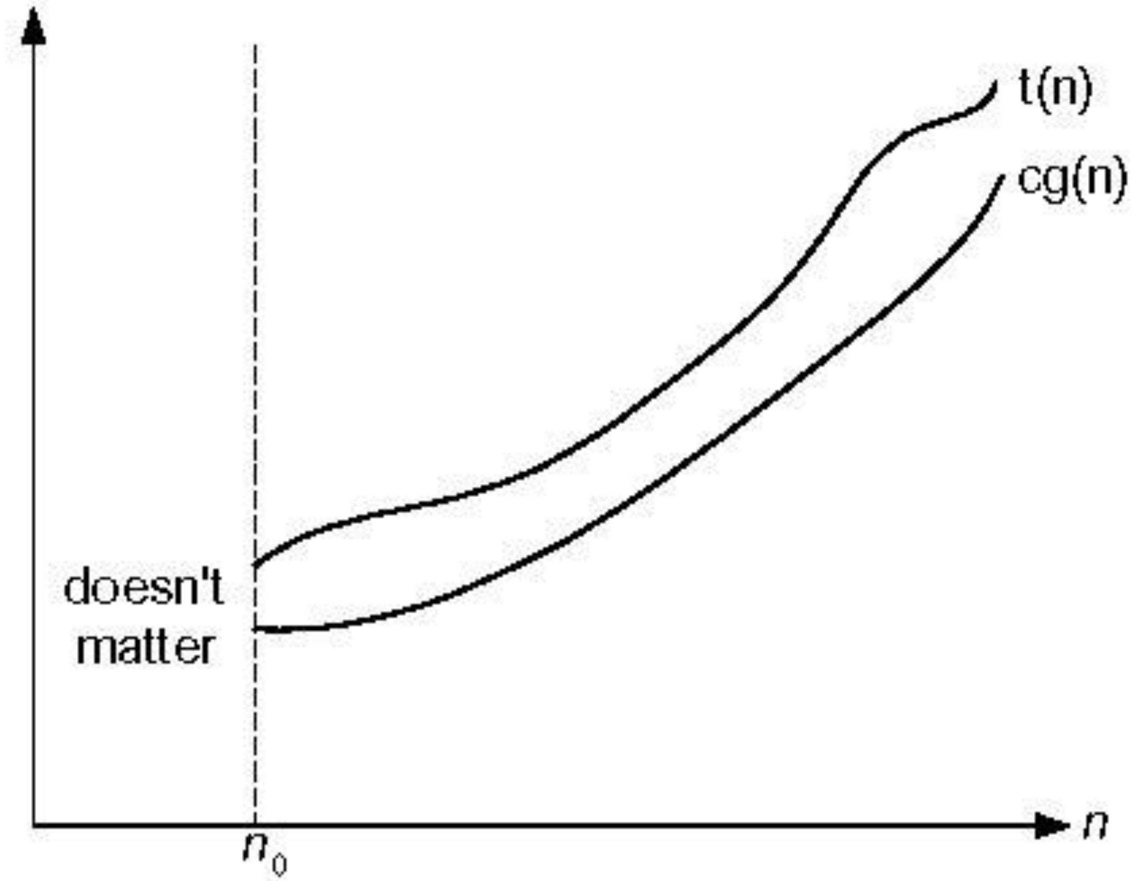


Fig. 2.2 Big-omega notation: $t(n) \in \Omega(g(n))$

Ω-notation

DEFINITION A function $t(n)$ is said to be in $\Omega(g(n))$, denoted $t(n) \in \Omega(g(n))$, if $t(n)$ is bounded below by some positive constant multiple of $g(n)$ for all large n, i.e., if there exist some positive constant c and some nonnegative integer n_0 such that

$$t(n) \geq cg(n) \quad \text{for all } n \geq n_0.$$

Big-theta

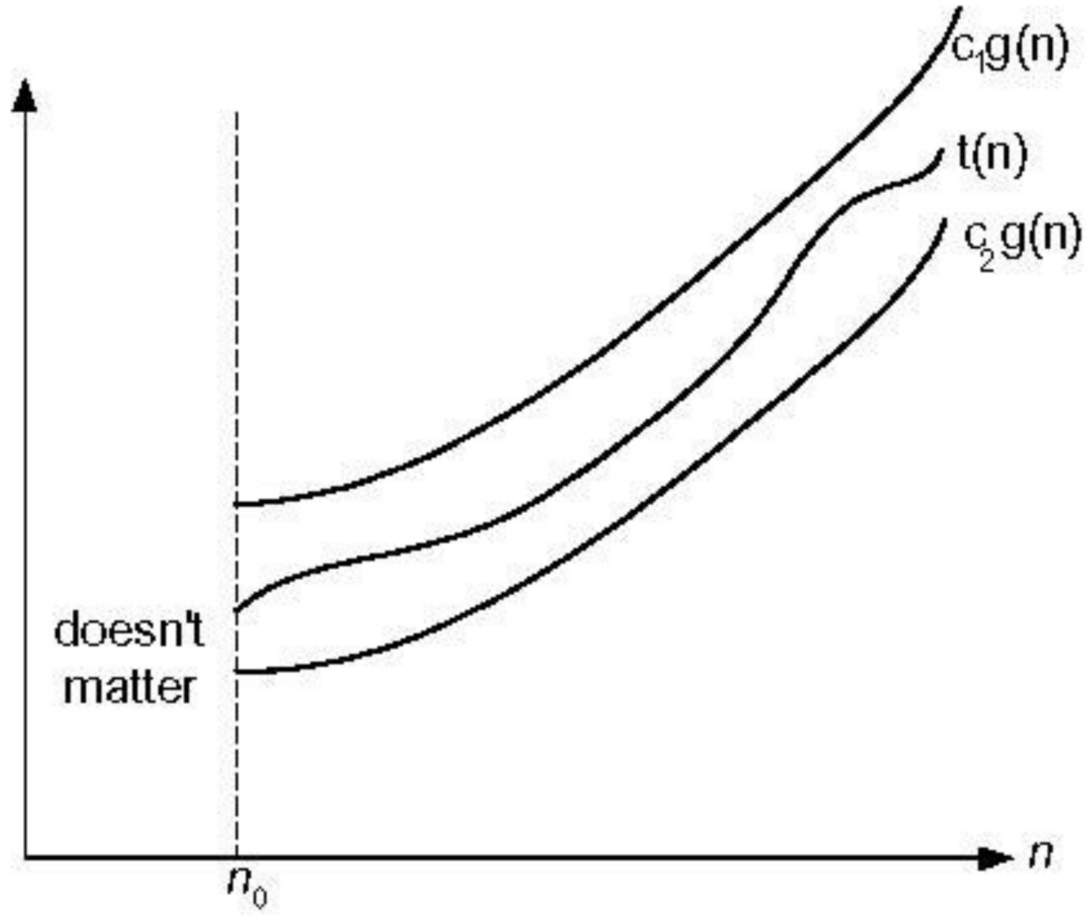


Figure 2.3 Big-theta notation: $t(n) \in \Theta(g(n))$

Θ-notation

DEFINITION A function $t(n)$ is said to be in $\Theta(g(n))$, denoted $t(n) \in \Theta(g(n))$, if $t(n)$ is bounded both above and below by some positive constant multiples of $g(n)$ for all large n, i.e., if there exist some positive constants c_1 and c_2 and some nonnegative integer n_0 such that

$$c_2 g(n) \le t(n) \le c_1 g(n) \quad \text{for all } n \ge n_0.$$

Establishing order of growth using the definition

Definition: $f(n)$ **is in O(**$g(n)$**) if order of growth of** $f(n) \le$ **order of growth of** $g(n)$ **(within constant multiple),**
i.e., there exist positive constant c **and non-negative integer** n_0 **such that**

$$f(n) \le c\, g(n) \text{ for every } n \ge n_0$$

Examples:

- $10n$ is $O(n^2)$

- $5n+20$ is $O(n)$

Some properties of asymptotic order of growth

- $f(n) \in O(f(n))$

- $f(n) \in O(g(n))$ iff $g(n) \in \Omega(f(\mathrm{n}))$

- If $f(n) \in O(g(n))$ and $g(n) \in O(h(n))$, then $f(n) \in O(h(n))$

  Note similarity with $a \leq \mathrm{b}$

- If $f_1(n) \in O(g_1(n))$ and $f_2(n) \in O(g_2(n))$, then

$$f_1(n) + f_2(n) \in O(\max\{g_1(n), g_2(n)\})$$

Example 1.11 The function $3n + 2 = O(n)$ as $3n + 2 \leq 4n$ for all $n \geq 2$. $3n + 3 = O(n)$ as $3n + 3 \leq 4n$ for all $n \geq 3$. $100n + 6 = O(n)$ as $100n + 6 \leq 101n$ for all $n \geq 6$. $10n^2 + 4n + 2 = O(n^2)$ as $10n^2 + 4n + 2 \leq 11n^2$ for all $n \geq 5$. $1000n^2 + 100n - 6 = O(n^2)$ as $1000n^2 + 100n - 6 \leq 1001n^2$ for $n \geq 100$. $6 * 2^n + n^2 = O(2^n)$ as $6 * 2^n + n^2 \leq 7 * 2^n$ for $n \geq 4$. $3n + 3 = O(n^2)$ as $3n + 3 \leq 3n^2$ for $n \geq 2$. $10n^2 + 4n + 2 = O(n^4)$ as $10n^2 + 4n + 2 \leq 10n^4$ for $n \geq 2$. $3n + 2 \neq O(1)$ as $3n + 2$ is not less than or equal to c for any constant c and all $n \geq n_0$. $10n^2 + 4n + 2 \neq O(n)$. □

Example 1.12 The function $3n+2 = \Omega(n)$ as $3n+2 \geq 3n$ for $n \geq 1$ (the inequality holds for $n \geq 0$, but the definition of Ω requires an $n_0 > 0$). $3n+3 = \Omega(n)$ as $3n+3 \geq 3n$ for $n \geq 1$. $100n+6 = \Omega(n)$ as $100n+6 \geq 100n$ for $n \geq 1$. $10n^2+4n+2 = \Omega(n^2)$ as $10n^2+4n+2 \geq n^2$ for $n \geq 1$. $6*2^n+n^2 = \Omega(2^n)$ as $6*2^n+n^2 \geq 2^n$ for $n \geq 1$. Observe also that $3n+3 = \Omega(1)$, $10n^2+4n+2 = \Omega(n)$, $10n^2+4n+2 = \Omega(1)$, $6*2^n+n^2 = \Omega(n^{100})$, $6*2^n+n^2 = \Omega(n^{50.2})$, $6*2^n+n^2 = \Omega(n^2)$, $6*2^n+n^2 = \Omega(n)$, and $6*2^n+n^2 = \Omega(1)$. □

Example 1.13 The function $3n + 2 = \Theta(n)$ as $3n + 2 \geq 3n$ for all $n \geq 2$ and $3n + 2 \leq 4n$ for all $n \geq 2$, so $c_1 = 3$, $c_2 = 4$, and $n_0 = 2$. $3n + 3 = \Theta(n)$, $10n^2 + 4n + 2 = \Theta(n^2)$, $6 * 2^n + n^2 = \Theta(2^n)$, and $10 * \log n + 4 = \Theta(\log n)$. $3n + 2 \neq \Theta(1)$, $3n + 3 \neq \Theta(n^2)$, $10n^2 + 4n + 2 \neq \Theta(n)$, $10n^2 + 4n + 2 \neq \Theta(1)$, $6 * 2^n + n^2 \neq \Theta(n^2)$, $6 * 2^n + n^2 \neq \Theta(n^{100})$, and $6 * 2^n + n^2 \neq \Theta(1)$. □

Comparisons

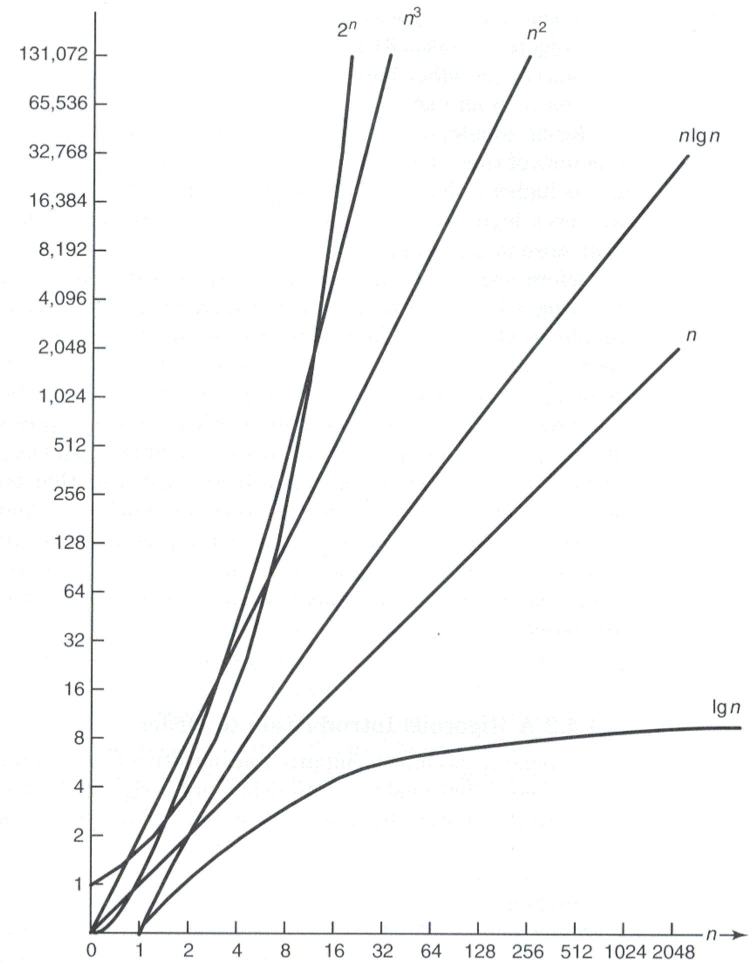
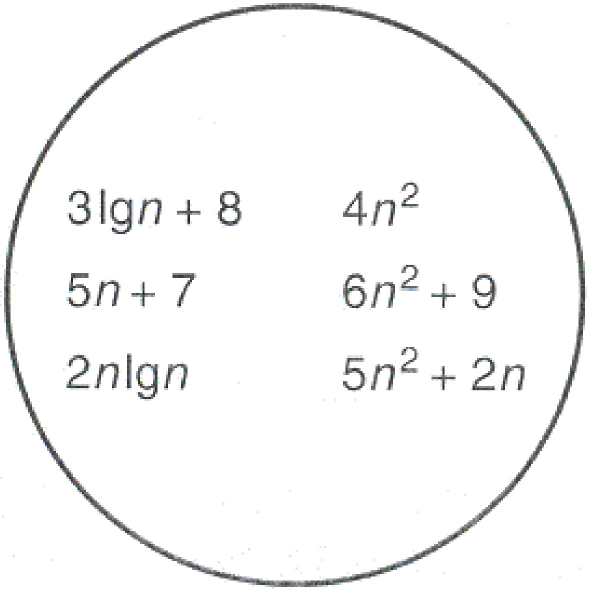
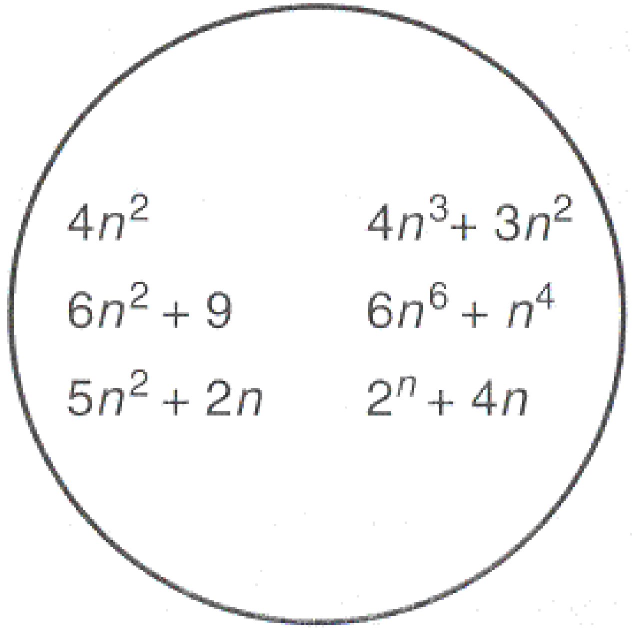


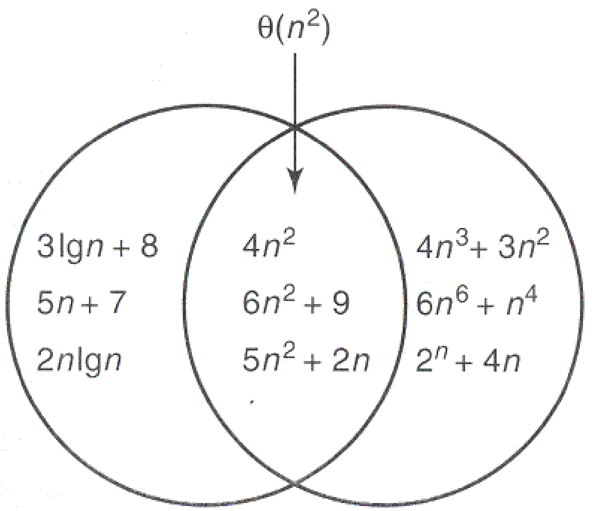
Figure 1.3 ● Growth rates of some common complexity functions.



(a) $O(n^2)$



(b) $\Omega(n^2)$



(c) $\theta(n^2) = O(n^2) \cap \Omega(n^2)$

Establishing order of growth using limits

$$\lim_{n\to\infty} T(n)/g(n) = \begin{cases} 0 & \text{order of growth of } T(n) < \text{ order of growth of } g(n) \\ c > 0 & \text{order of growth of } T(n) = \text{ order of growth of } g(n) \\ \infty & \text{order of growth of } T(n) > \text{ order of growth of } g(n) \end{cases}$$

Examples:

- $10n$ vs. n^2
- $n(n+1)/2$ vs. n^2

L'Hôpital's rule and Stirling's formula

L'Hôpital's rule: If $\lim_{n\to\infty} f(n) = \lim_{n\to\infty} g(n) = \infty$ and the derivatives f', g' exist, then

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

Example: $\log n$ vs. n

Stirling's formula: $n! \approx (2\pi n)^{1/2} (n/e)^n$

Example: 2^n vs. $n!$

Orders of growth of some important functions

- **All logarithmic functions $\log_a n$ belong to the same class $\Theta(\log n)$ no matter what the logarithm's base $a > 1$ is**

- **All polynomials of the same degree k belong to the same class: $a_k n^k + a_{k-1} n^{k-1} + \ldots + a_0 \in \Theta(n^k)$**

- **Exponential functions a^n have different orders of growth for different a's**

- **order $\log n$ < order n^α (α>0) < order a^n < order $n!$ < order n^n**

Definition 1.7 [Little "oh"] The function $f(n) = o(g(n))$ (read as "f of n is little oh of g of n") iff

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

□

Definition 1.8 [Little omega] The function $f(n) = \omega(g(n))$ (read as "f of n is little omega of g of n") iff

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

Basic asymptotic efficiency classes

1	constant
$\log n$	logarithmic
n	linear
$n \log n$	n-log-n
n^2	quadratic
n^3	cubic
2^n	exponential
$n!$	factorial

EXAMPLE 1 Compare the orders of growth of $\frac{1}{2}n(n-1)$ and n^2. (This is one of the examples we used at the beginning of this section to illustrate the definitions.)

$$\lim_{n\to\infty} \frac{\frac{1}{2}n(n-1)}{n^2} = \frac{1}{2} \lim_{n\to\infty} \frac{n^2 - n}{n^2} = \frac{1}{2} \lim_{n\to\infty} (1 - \frac{1}{n}) = \frac{1}{2}.$$

Since the limit is equal to a positive constant, the functions have the same order of growth or, symbolically, $\frac{1}{2}n(n-1) \in \Theta(n^2)$. ■

EXAMPLE 2 Compare the orders of growth of $\log_2 n$ and $\sqrt{n}$. (Unlike Example 1, the answer here is not immediately obvious.)

$$\lim_{n\to\infty} \frac{\log_2 n}{\sqrt{n}} = \lim_{n\to\infty} \frac{(\log_2 n)'}{(\sqrt{n})'} = \lim_{n\to\infty} \frac{(\log_2 e)\frac{1}{n}}{\frac{1}{2\sqrt{n}}} = 2\log_2 e \lim_{n\to\infty} \frac{1}{\sqrt{n}} = 0.$$

Since the limit is equal to zero, $\log_2 n$ has a smaller order of growth than $\sqrt{n}$. (Since $\lim_{n\to\infty} \frac{\log_2 n}{\sqrt{n}} = 0$, we can use the so-called ***little-oh notation***: $\log_2 n \in o(\sqrt{n})$.

EXAMPLE 3 Compare the orders of growth of $n!$ and 2^n. (We discussed this informally in Section 2.1.) Taking advantage of Stirling's formula, we get

$$\lim_{n\to\infty} \frac{n!}{2^n} = \lim_{n\to\infty} \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n}{2^n} = \lim_{n\to\infty} \sqrt{2\pi n}\,\frac{n^n}{2^n e^n} = \lim_{n\to\infty} \sqrt{2\pi n}\left(\frac{n}{2e}\right)^n = \infty.$$

Check the assertions?

a. $n(n+1)/2 \in O(n^3)$

b. $n(n+1)/2 \in O(n^2)$

c. $n(n+1)/2 \in \Theta(n^3)$

d. $n(n+1)/2 \in \Omega(n)$

a. $n(n+1)/2 \in O(n^3)$ is true.

b. $n(n+1)/2 \in O(n^2)$ is true.

c. $n(n+1)/2 \in \Theta(n^3)$ is false.

d. $n(n+1)/2 \in \Omega(n)$ is true.

3. For each of the following functions, indicate the class $\Theta(g(n))$ the function belongs to. (Use the simplest $g(n)$ possible in your answers.) Prove your assertions.

a. $(n^2+1)^{10}$

b. $\sqrt{10n^2+7n+3}$

c. $2n\lg(n+2)^2+(n+2)^2\lg\frac{n}{2}$

d. $2^{n+1}+3^{n-1}$

e. $\lfloor\log_2 n\rfloor$

$$\lim_{n\to\infty} \frac{(n^2+1)^{10}}{n^{20}} = \lim_{n\to\infty} \frac{(n^2+1)^{10}}{(n^2)^{10}} = \lim_{n\to\infty} \left(\frac{n^2+1}{n^2}\right)^{10} == \lim_{n\to\infty} \left(1 + \frac{1}{n^2}\right)^{10} = 1.$$

Hence $(n^2+1)^{10} \in \Theta(n^{20})$.

Note: An alternative proof can be based on the binomial formula and the assertion of Exercise 6a.

b. Informally, $\sqrt{10n^2+7n+3} \approx \sqrt{10n^2} = \sqrt{10}n \in \Theta(n)$. Formally,

$$\lim_{n\to\infty} \frac{\sqrt{10n^2+7n+3}}{n} = \lim_{n\to\infty} \sqrt{\frac{10n^2+7n+3}{n^2}} = \lim_{n\to\infty} \sqrt{10 + \frac{7}{n} + \frac{3}{n^2}} = \sqrt{10}.$$

Hence $\sqrt{10n^2+7n+3} \in \Theta(n)$.

c. $2n\lg(n+2)^2 + (n+2)^2\lg\frac{n}{2} = 2n2\lg(n+2) + (n+2)^2(\lg n - 1) \in \Theta(n\lg n) + \Theta(n^2\lg n) = \Theta(n^2\lg n)$.

d. $2^{n+1} + 3^{n-1} = 2^n 2 + 3^n \frac{1}{3} \in \Theta(2^n) + \Theta(3^n) = \Theta(3^n)$.

Prove that it is in increasing order

$$\log n, \quad n, \quad n\log n, \quad n^2, \quad n^3, \quad 2^n, \quad n!$$

Magic Square

15	8	1	24	17
16	14	7	5	23
22	20	13	6	4
3	21	19	12	10
9	2	25	18	11

Maximum Rule

Consider an algorithm that proceeds in three steps: initialisation, processing and finalisation, and that these steps take time in $\Theta(n^2)$, $\Theta(n^3)$ and $\Theta(n \log n)$ respectively. It is therefore clear that the complete algorithm takes a time in $\Theta(n^2 + n^3 + n \log n)$. From the maximum rule

$$
\begin{aligned}
\Theta(n^2 + n^3 + n \log n) &= \Theta(\max(n^2, n^3 + n \log n)) \\
&= \Theta(\max(n^2, \max(n^3, n \log n))) \\
&= \Theta(\max(n^2, n^3)) \\
&= \Theta(n^3)
\end{aligned}
$$

Conditional asymptotic notation

- **Many algorithms easier to analyse if initially we restrict our attention to instances whose size satisfies a certain condition, such as a power of 2.**

Contd…

More generally, let $f, t : N \rightarrow R^{\geq 0}$ be two functions from the natural numbers to the nonnegative reals, and let $P : N \rightarrow \{true, false\}$ be a property of the integers. We say that $t(n)$ is in $O(f(n) \mid P(n))$ if $t(n)$ is bounded above by a positive real multiple of $f(n)$ for all sufficiently large n such that $P(n)$ holds. Formally, $O(f(n) \mid P(n))$ is defined as

Contd…

$$O(f(n) \mid P(n)) =$$

$$\{t : N \rightarrow R^{\geq 0} \mid \exists c \in R^{+} \ \exists n_0 \in N \ \forall n \geq n_0 \ (P(n) \Rightarrow t(n) \leq cf(n))\}$$

Contd…

The sets $\Omega(f(n) \mid P(n))$ and $\Theta(f(n) \mid P(n))$ are defined in a similar way.

Conditional asymptotic notation is more than a mere notational convenience: its main interest is that it can generally be eliminated once it has been used to facilitate the analysis of an algorithm. For this we need a few definitions. A function $f : N \rightarrow R^{\geq 0}$ is *eventually nondecreasing* if there exists an integer threshold n_0 such that $f(n) \leq f(n+1)$ for all $n \geq n_0$. This implies by mathematical induction that $f(n) \leq f(m)$ whenever $m \geq n \geq n_0$.

Contd…

Let $b \geq 2$ be any integer. Function f is *b-smooth* if, in addition to being eventually nondecreasing, it satisfies the condition $f(bn) \in O(f(n))$. In other words, there must exist a constant c (depending on b) such that $f(bn) \leq cf(n)$ for all $n \geq n_0$. A function is *smooth* if it is b-smooth for every integer $b \geq 2$.