Big O Notation

Runtime of an algorithm

Running time of an algorithm depends on

i) Input size

6	13	14	25	33

Array size = 5

6	13	14	25	33	43	51	53	64	72	84

Array size = 11

Runtime of an algorithm

Running time of an algorithm depends on

i) Input size

a function of the size of its input

f(n)

n = input size

6	13	14	25	33

Array size = 5

6	13	14	25	33	43	51	53	64	72	84

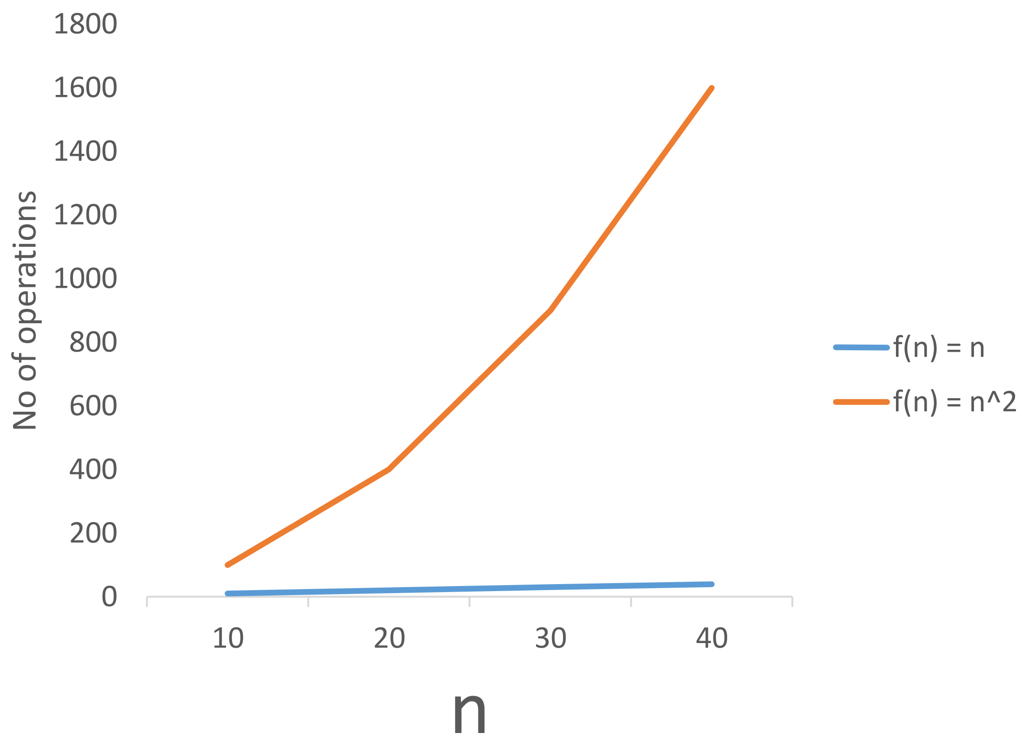
Array size = 11

Growth rate of a function

Rate of growth: How fast a function grows with the input size

f(n) = n

f(n) = n^2

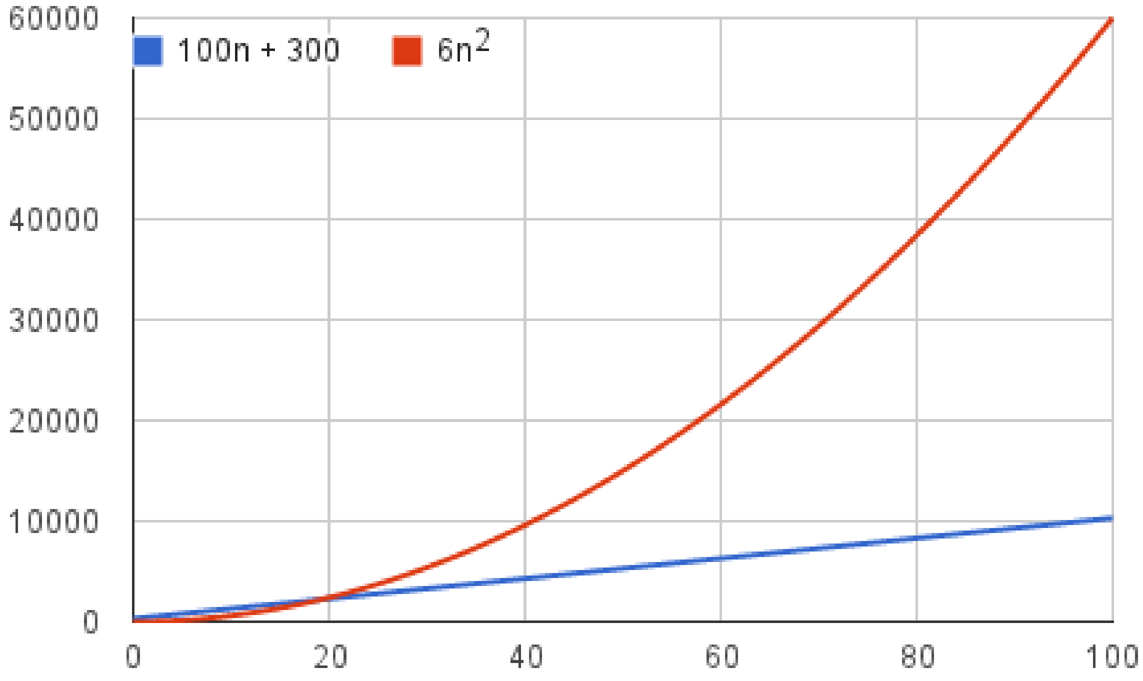


Asymptotic notation of a function

f(n) = $6n^2$ + 100n + 30

n >= 20

$6n^2$ > 100n + 30

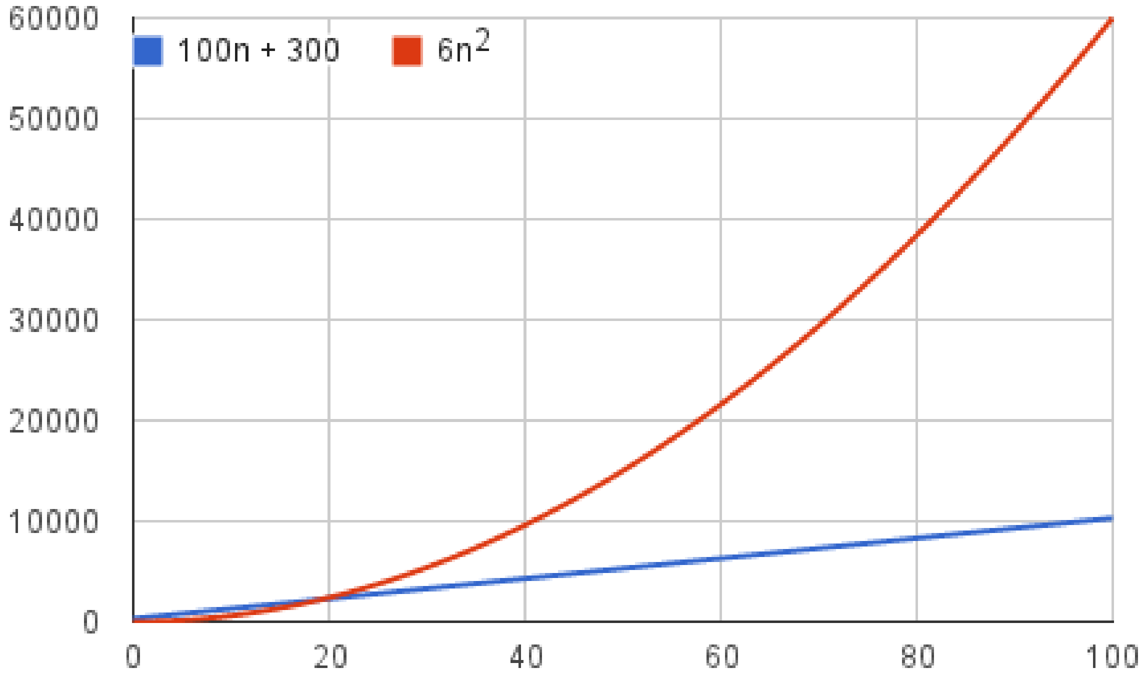


Asymptotic notation of a function

f(n) = $6n^2$ + 100n + 30

The algorithm grows as n^2

Drop coefficient 6 and terms 100n + 30. They are not significant enough.

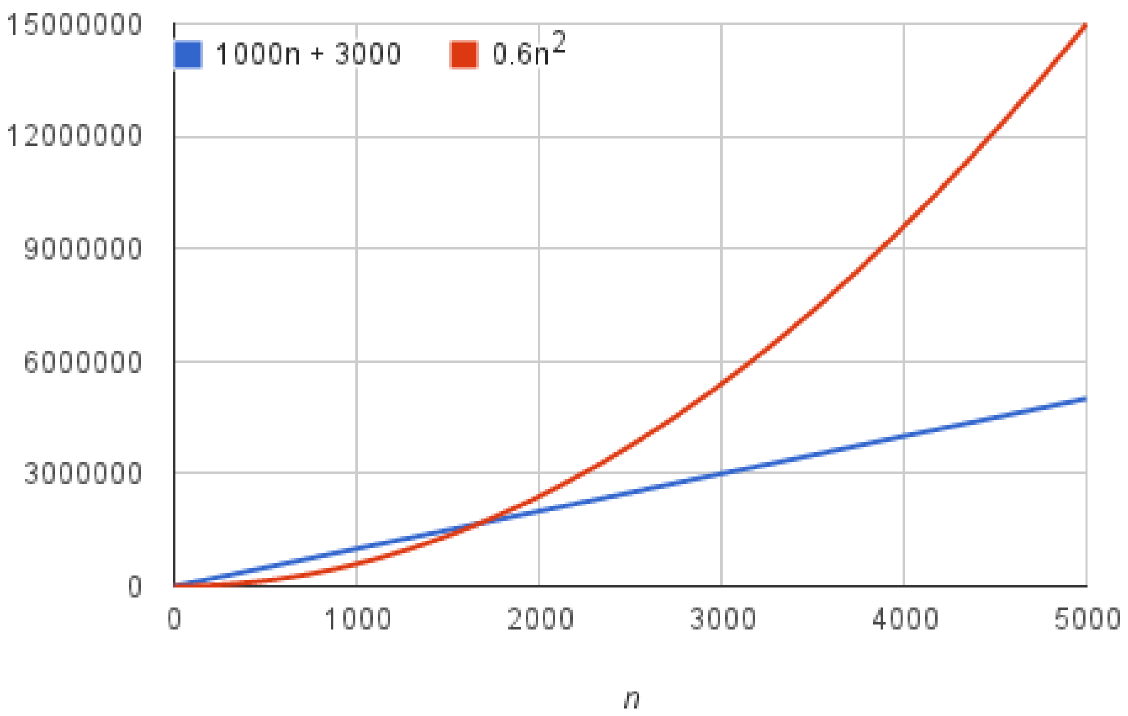


Asymptotic notation of a function

f(n) = $0.6n^2$ + 1000n + 3000

n >= 1000

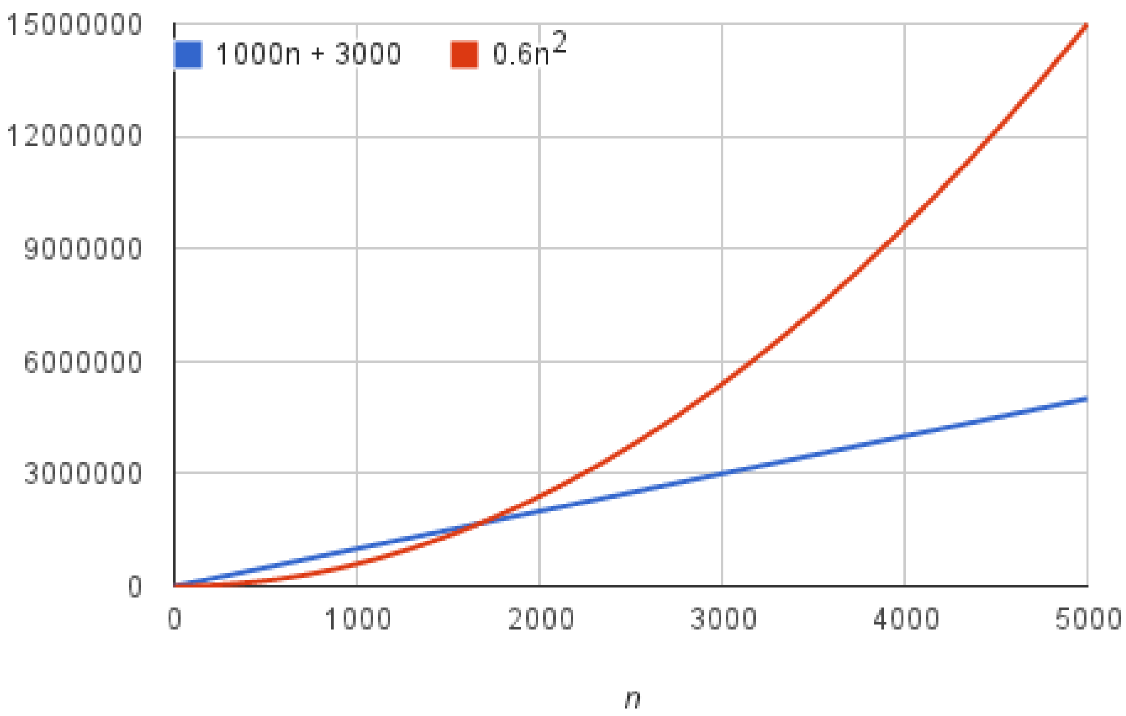
$0.6n^2$ > 1000n + 3000



Asymptotic notation of a function

f(n) = $0.6n^2$ + 1000n + 3000

The algorithm grows as n^2



Asymptotic notation of a function

f(n) = 0.6n^2 + 1000n + 3000

f(n) = 6n^2 + 100n + 30

These algorithm grows as n^2

When we drop the constant coefficients and the less significant terms, we use **asymptotic notation**

Big O Notation

Given f(n): the actual growth rate of your algorithm as a function of input size find g(n) such that

$$C|g(n)| >= |f(n)| \text{ for } n > k$$

Then f(n) = O (g(n))

Big O Notation

Find the Big O Notation of $n^2 + 2n + 1$

$$f(n) = n^2 + 2n + 1 \leq n^2 + 2n^2 + n^2$$

$$= 4n^2 \ = \mathrm{Cg(n)}$$

Find k:

n = 1, f(n) = 4, Cg(n) = 4

n = 2, f(n) = 9, Cg(n) = 16

For n > 1 Cg(n) >= f(n)

Hence, k = 1

f(n) = O(g(n)) = O(n^2),

Where k = 1, C = 4

Big O Notation

Find the Big O Notation of n!

n! = 1.2.3 … n <= n.n … n

= n^n = C g(n)

f(n) = O(g(n)) = O(n^n),

Where k = 1, C = 1

Find k:

n = 1, f(n) = 1, Cg(n) = 1

n = 2, f(n) = 2, Cg(n) = 4

For n > 1 Cg(n) >= f(n)

Hence, k = 1

Big O Notation

Find the Big O Notation of log(n!)

Big O Notation

If $f1(x) = O(g1(x))$ and $f2(x) = O(g2(x))$

$$(f1 + f2)(x) = O(\max(g1(x), g2(x)))$$

$$(f1f2)(x) = O(g1(x)g2(x))$$

Big O Notation

Find the Big O Notation of $f(n) = 3n\log(n!) + (n^2 + 3)\log(n)$

Big O Notation

Find the Big O Notation of f(n) = 3nlog(n!) + (n^2+ 3)log(n)

Common Big O Notation

Notation	Name
O(1)	Constant
O(logn)	Logarithmic
$O((\log n)^c)$	Poly-logarithmic
O(n)	Linear
$O(n^2)$	Quadratic
$O(n^c)$	Polynomial
$O(c^n)$	Exponential