Design & Analysis of Algorithms

CS11313 - Fall 2020

Day # 2 and 3

The Big-*O* Notation and its Relatives

Today's Agenda

- Running Time Orders of Growth.
- A formal definition of Big-O
- Big-O Relatives

Orders of Growth (Review)

- Order of Growth of the running time: How quickly the running time of an algorithm grows as the input size grows.
  Examples: $\log n, \quad n, \quad n^2, \quad n^3, \quad 2^n,$ etc.

Orders of Growth (Review)

- Order of Growth of the running time: How quickly the running time of an algorithm grows as the input size grows.
  Examples: $\log n, \quad n, \quad n^2, \quad n^3, \quad 2^n,$ etc.

- Focus on the highest order term:

Orders of Growth (Review)

- Order of Growth of the running time: How quickly the running time of an algorithm grows as the input size grows.
  Examples: $\log n$, n, n^2, n^3, 2^n, etc.

- Focus on the highest order term:
  - Example: $n^2 + n + \log n$ is in the order of n^2.

Orders of Growth (Review)

- Order of Growth of the running time: How quickly the running time of an algorithm grows as the input size grows.
  Examples: $\log n$, n, n^2, n^3, 2^n, etc.

- Focus on the highest order term:
  - Example: $n^2 + n + \log n$ is in the order of n^2.
  - Rationale: When n becomes large, time due to the lower order terms becomes insignificant compared to the highest order term.

- Drop the coefficient of the highest order term:

Orders of Growth (Review)

- Order of Growth of the running time: How quickly the running time of an algorithm grows as the input size grows.
  Examples: $\log n,\ \ n,\ \ n^2,\ \ n^3,\ \ 2^n,$ etc.

- Focus on the highest order term:
  - Example: $n^2 + n + \log n$ is in the order of n^2.
  - Rationale: When n becomes large, time due to the lower order terms becomes insignificant compared to the highest order term.

- Drop the coefficient of the highest order term:
  - Example: $n^2,\ \ \frac{1}{2}n^2$ and $10n^2$ are all in the order of n^2.

Orders of Growth (Review)

- Order of Growth of the running time: How quickly the running time of an algorithm grows as the input size grows.
  Examples: $\log n$, n, n^2, n^3, 2^n, etc.

- Focus on the highest order term:
  - Example: $n^2 + n + \log n$ is in the order of n^2.
  - Rationale: When n becomes large, time due to the lower order terms becomes insignificant compared to the highest order term.

- Drop the coefficient of the highest order term:
  - Example: n^2, $\frac{1}{2}n^2$ and $10n^2$ are all in the order of n^2.
  - Rationale:
    - Quadratic growth is not the same as, linear or cubic growth, etc.

Orders of Growth (Review)

- Order of Growth of the running time: How quickly the running time of an algorithm grows as the input size grows.
  Examples: $\log n$, n, n^2, n^3, 2^n, etc.

- Focus on the highest order term:
  - Example: $n^2 + n + \log n$ is in the order of n^2.
  - Rationale: When n becomes large, time due to the lower order terms becomes insignificant compared to the highest order term.

- Drop the coefficient of the highest order term:
  - Example: n^2, $\frac{1}{2}n^2$ and $10n^2$ are all in the order of n^2.
  - Rationale:
    - Quadratic growth is not the same as, linear or cubic growth, etc.
    - Algorithms have different constants when implemented, based on hardware, software and implementation factors.

Orders of Growth (Review)

- Order of Growth of the running time: How quickly the running time of an algorithm grows as the input size grows.
  Examples: $\log n, \quad n, \quad n^2, \quad n^3, \quad 2^n,$ etc.

- Focus on the highest order term:
  - Example: $n^2 + n + \log n$ is in the order of n^2.
  - Rationale: When n becomes large, time due to the lower order terms becomes insignificant compared to the highest order term.

- Drop the coefficient of the highest order term:
  - Example: $n^2, \quad \frac{1}{2}n^2$ and $10n^2$ are all in the order of n^2.
  - Rationale:
    - Quadratic growth is not the same as, linear or cubic growth, etc.
    - Algorithms have different constants when implemented, based on hardware, software and implementation factors.

! Assumption is good for theoretical study of algorithms, not fo practical comparison between algorithms (*stay tuned*).

Quiz # 1

Assume $T(n)$ is the order of growth of the running time of Bubble Sort as a function of the input size n. Which of the following is *true* about $T(n)$?

- **A.** $T(n) = O(n^2)$
- **B.** $T(n) = O(n^3)$
- **C.** $T(n) = O(2^n)$
- **D.** All of the above.
- **E.** None of the above.

What is

Big-*O* anyway?

Big-O

Definition. Let $f(n)$ and $g(n)$ be two functions that are always positive, $f(n)$ is said to be $O(g)$ if and only if :

> There are two constants c and n_o , such that
> $0 \leq f(n) \leq c \bullet g(n)$ for all $n \geq n_o$

Big-O

Definition. Let $f(n)$ and $g(n)$ be two functions that are always positive, $f(n)$ is said to be $O(g)$ if and only if :

There are two constants c and n_o , such that
$0 \leq f(n) \leq c \bullet g(n)$ for all $n \geq n_o$

Less formally: If multiplying $g(n)$ by a constant makes it an upper bound for $f(n)$ after some point, then f is $O(g)$.

Example # 1

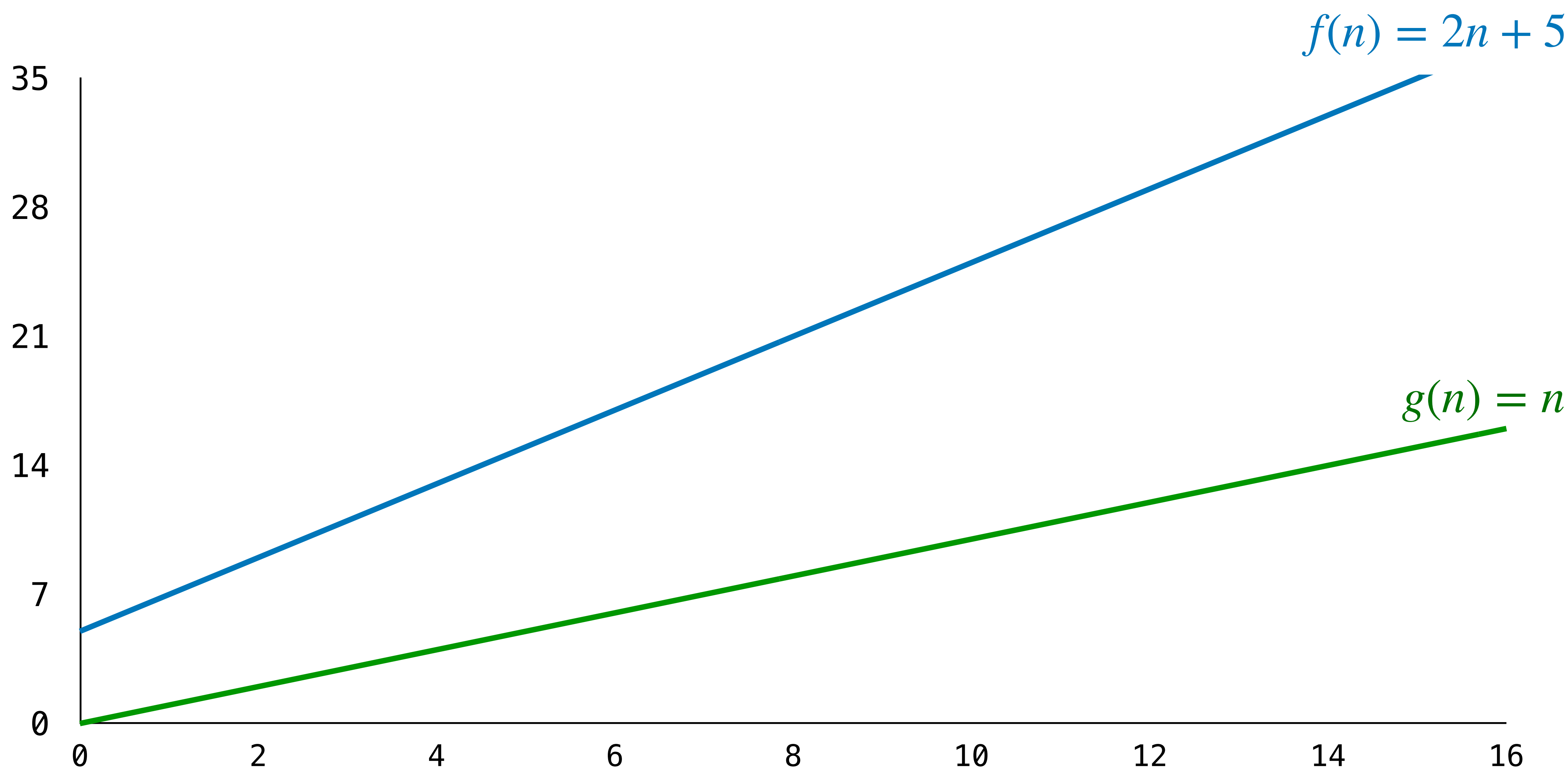
Assume $f(n) = 2n + 5$ and $g(n) = n$.

Example # 1

Assume $f(n) = 2n + 5$ and $g(n) = n$.

f is $O(g)$ because there are c and n_o such that $0 \leq f(n) \leq c \bullet g(n)$ for all $n \geq n_o$:

If $c = 3$, then $3 \bullet g(n) \geq f(n)$ for all $n \geq 5$

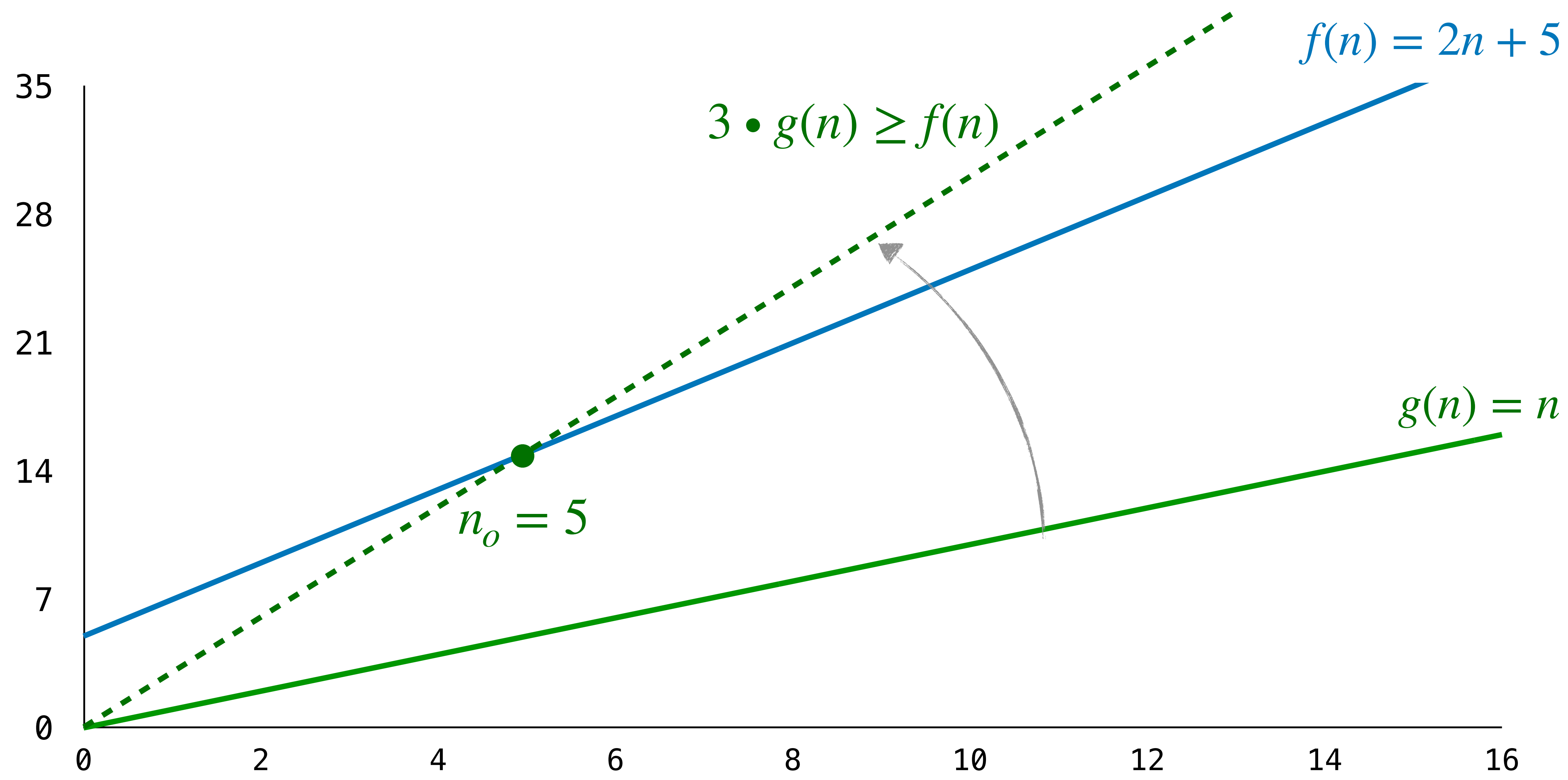


Example # 1

Assume $f(n) = 2n + 5$ and $g(n) = n$.

f is $O(g)$ because there are c and n_o such that $0 \le f(n) \le c \bullet g(n)$ for all $n \ge n_o$:

If $c = 3$, then $3 \bullet g(n) \ge f(n)$ for all $n \ge 5$



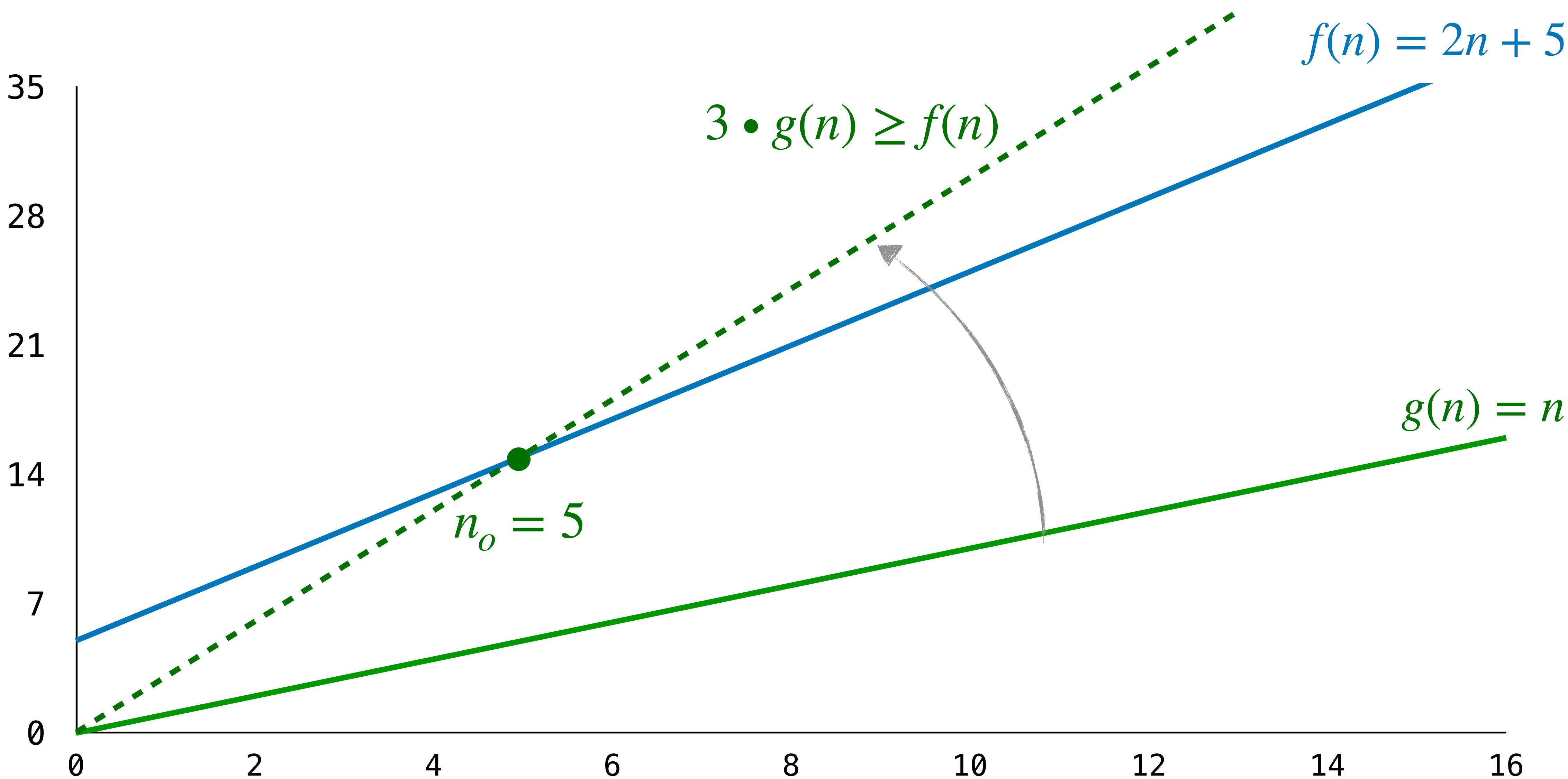
Example # 1

Assume $f(n) = 2n + 5$ and $g(n) = n$.

f is $O(g)$ because there are c and n_o such that $0 \le f(n) \le c \bullet g(n)$ for all $n \ge n_o$:

If $c = 3$, then $3 \bullet g(n) \ge f(n)$ for all $n \ge 5$

not the only possible c and n_o



Example # 2

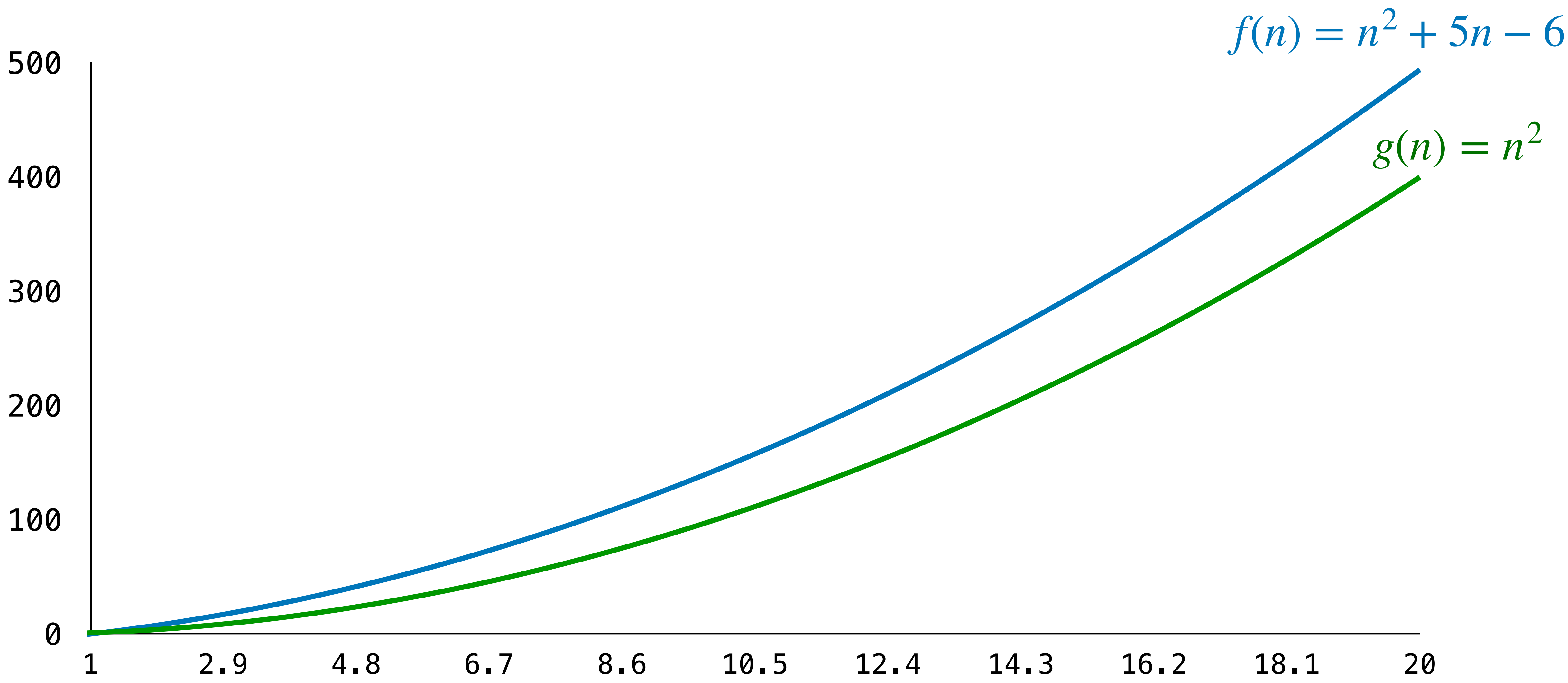
Assume $f(n) = n^2 + 5n - 6$ and $g(n) = n^2$.

Example # 2

Assume $f(n) = n^2 + 5n - 6$ and $g(n) = n^2$.

f is $O(g)$ because there are c and n_o such that $0 \leq f(n) \leq c \bullet g(n)$ for all $n \geq n_o$:

If $c = 7$, then $7 \bullet g(n) \geq f(n)$ for all $n \geq 1$

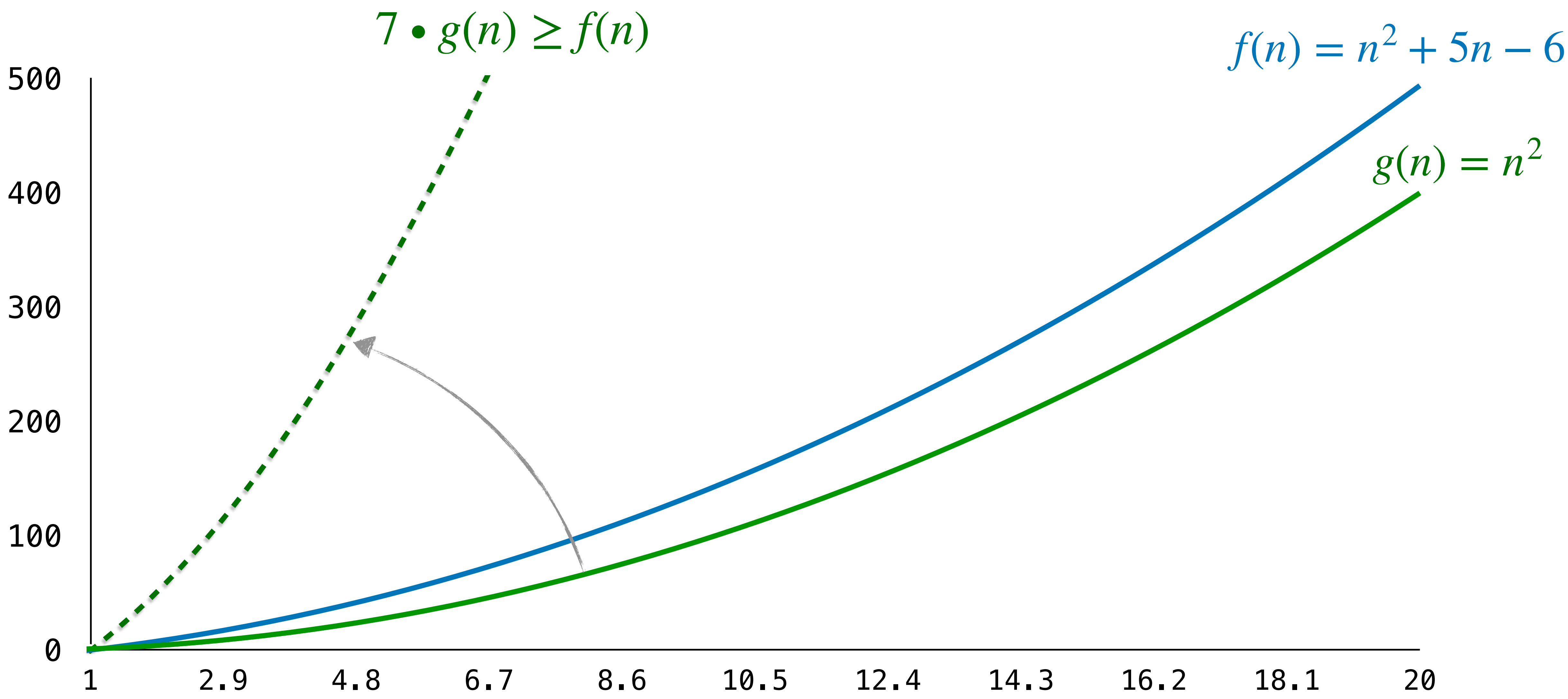


Example # 2

Assume $f(n) = n^2 + 5n - 6$ and $g(n) = n^2$.

f is $O(g)$ because there are c and n_o such that $0 \leq f(n) \leq c \bullet g(n)$ for all $n \geq n_o$:

If $c = 7$, then $7 \bullet g(n) \geq f(n)$ for all $n \geq 1$



Exercise # 1

For each of the following function, show that f is $O(n)$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3 \quad \text{and} \quad g(n) = n$

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \quad \leq \quad 3n + 3n \quad$ for all $n \geq 1$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \quad \leq \quad 3n + 3n \quad \leq \quad 6n$ for all $n \geq 1$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$$\underbrace{3n + 3}_{f(n)} \leq 3n + 3n \leq 6n \leq \underbrace{9n}_{c \bullet f(n)} \text{ for all } n \geq 1.$$

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \le c \bullet g(n)$ for all $n \ge 1$.

$3n + 3 \quad \le \quad 3n + 3n \quad \le \quad 6n \le \quad 9n$ for all $n \ge 1$.

B. $f(n) = n^2 + 5n - 6$ and $g(n) = n^2$

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \quad \leq \quad 3n + 3n \quad \leq \quad 6n \leq \quad 9n$ for all $n \geq 1$.

B. $f(n) = n^2 + 5n - 6$ and $g(n) = n^2$

Solution.

If we pick $c = 7$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \leq 3n + 3n \leq 6n \leq 9n$ for all $n \geq 1$.

B. $f(n) = n^2 + 5n - 6$ and $g(n) = n^2$

Solution.

If we pick $c = 7$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$n^2 + 5n - 6 \leq n^2 + 5n$ for all $n \geq 1$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \quad \leq \quad 3n + 3n \quad \leq \quad 6n \leq \quad 9n \quad$ for all $n \geq 1$.

B. $f(n) = n^2 + 5n - 6$ and $g(n) = n^2$

Solution.

If we pick $c = 7$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$n^2 + 5n - 6 \quad \leq \quad n^2 + 5n \quad \leq \quad n^2 + 5n^2 \quad$ for all $n \geq 1$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \quad \leq \quad 3n + 3n \quad \leq \quad 6n \leq \quad 9n$ for all $n \geq 1$.

B. $f(n) = n^2 + 5n - 6$ and $g(n) = n^2$

Solution.

If we pick $c = 7$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$n^2 + 5n - 6 \quad \leq \quad n^2 + 5n \quad \leq \quad n^2 + 5n^2 \leq \quad 6n^2$ for all $n \geq 1$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \leq 3n + 3n \leq 6n \leq 9n$ for all $n \geq 1$.

B. $f(n) = n^2 + 5n - 6$ and $g(n) = n^2$

Solution.

If we pick $c = 7$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$$\underbrace{n^2 + 5n - 6}_{f(n)} \leq n^2 + 5n \leq n^2 + 5n^2 \leq 6n^2 \leq \underbrace{7n^2}_{c \bullet f(n)} \quad \text{for all } n \geq 1.$$

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3$ and $g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \quad \leq \quad 3n + 3n \quad \leq \quad 6n \leq \quad 9n \quad$ for all $n \geq 1$.

B. $f(n) = n^2 + 5n - 6$ and $g(n) = n^2$

Solution.

If we pick $c = 7$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \quad \leq \quad 3n + 3n \quad \leq \quad 6n \leq \quad 9n \quad$ for all $n \geq 1$.

C. $f(n) = n^2$ and $g(n) = n^3$

Exercise # 1

For each of the following function, show that f is $O(n)$.

A. $f(n) = 3n + 3 \quad \text{and} \quad g(n) = n$

Solution.

If we pick $c = 9$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \quad \leq \quad 3n + 3n \quad \leq \quad 6n \leq \quad 9n \quad \text{for all } n \geq 1.$

B. $f(n) = n^2 + 5n - 6 \quad \text{and} \quad g(n) = n^2$

Solution.

If we pick $c = 7$, we can show that $f(n) \leq c \bullet g(n)$ for all $n \geq 1$.

$3n + 3 \quad \leq \quad 3n + 3n \quad \leq \quad 6n \leq \quad 9n \quad \text{for all } n \geq 1.$

C. $f(n) = n^2 \quad \text{and} \quad g(n) = n^3$

Solution.

If we pick $c = 1$, It is clear that $c \bullet g(n) \geq f(n)$ for all $n \geq 1$.

$n^2 \quad \leq \quad n^3 \quad \text{for all } n \geq 1.$

Exercise # 1

For each of the following function, show that f is $O(n)$.

D. $f(n) = 2^n$ and $g(n) = 3^n$

Exercise # 1

For each of the following function, show that f is $O(n)$.

D. $f(n) = 2^n$ and $g(n) = 3^n$

Solution.

We need to show that: $2^n \leq c \bullet 3^n$ for all $n \geq n_o$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

D. $f(n) = 2^n$ and $g(n) = 3^n$

Solution.

We need to show that: $2^n \leq c \bullet 3^n$ for all $n \geq n_o$.

Divide both sides by 2^n: $1 \leq c \bullet (\frac{3}{2})^n$

Exercise # 1

For each of the following function, show that f is $O(n)$.

D. $f(n) = 2^n$ and $g(n) = 3^n$

Solution.

We need to show that: $2^n \leq c \cdot 3^n$ for all $n \geq n_o$.

Divide both sides by 2^n: $1 \leq c \cdot (\frac{3}{2})^n$

The left hand side is constant, whereas the right hand side is an increasing function. Therefore, there must be constants n_o and c such that $1 \leq c \cdot (\frac{3}{2})^n$ for all $n \geq n_o$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

E. $f(n) = k^n$ and $g(n) = n!$ where k is a constant.

Exercise # 1

For each of the following function, show that f is $O(n)$.

E. $f(n) = k^n$ and $g(n) = n!$ where k is a constant.

Solution.

We need to show that: $k^n \leq c \bullet n!$ for all $n \geq n_o$.

Exercise # 1

For each of the following function, show that f is $O(n)$.

E. $f(n) = k^n$ and $g(n) = n!$ where k is a constant.

Solution.

We need to show that: $k^n \leq c \bullet n!$ for all $n \geq n_o$.

Take the $\log_2$ of both sides: $\log_2(k^n) \leq \log_2(c \bullet n!)$

Exercise # 1

For each of the following function, show that f is $O(n)$.

E. $f(n) = k^n$ and $g(n) = n!$ where k is a constant.

Solution.

We need to show that: $k^n \leq c \bullet n!$ for all $n \geq n_o$.

Take the $\log_2$ of both sides: $\log_2(k^n) \leq \log_2(c \bullet n!)$

$$n \log_2 k \leq \log_2 c + \log_2(n!)$$

Exercise # 1

For each of the following function, show that f is $O(n)$.

E. $f(n) = k^n$ and $g(n) = n!$ where k is a constant.

Solution.

We need to show that: $k^n \leq c \bullet n!$ for all $n \geq n_o$.

Take the $\log_2$ of both sides: $\log_2(k^n) \leq \log_2(c \bullet n!)$

$$n \log_2 k \leq \log_2 c + \log_2(n!)$$

$$n \log_2 k \leq \log_2 c + \underline{n \log_2 n - n \log_2 e + r \log_2 n}$$

Stirling's Approximation
(r is a positive constant)

Exercise # 1

For each of the following function, show that f is $O(n)$.

E. $f(n) = k^n$ and $g(n) = n!$ where k is a constant.

Solution.

We need to show that: $k^n \leq c \bullet n!$ for all $n \geq n_o$.

Take the $\log_2$ of both sides:
$$\log_2(k^n) \leq \log_2(c \bullet n!)$$

$$n \log_2 k \leq \log_2 c + \log_2(n!)$$

$$n \log_2 k \leq \log_2 c + n \log_2 n - n \log_2 e + r \log_2 n$$

Divide both sides by n:
$$\log_2 k \leq \frac{\log_2 c}{n} + \log_2 n - \log_2 e + \frac{r}{n} \log_2 n$$

Exercise # 1

For each of the following function, show that f is $O(n)$.

E. $f(n) = k^n$ and $g(n) = n!$ where k is a constant.

Solution.

We need to show that: $k^n \leq c \bullet n!$ for all $n \geq n_o$.

Take the $\log_2$ of both sides: $\log_2(k^n) \leq \log_2(c \bullet n!)$

$$n \log_2 k \leq \log_2 c + \log_2(n!)$$

$$n \log_2 k \leq \log_2 c + n \log_2 n - n \log_2 e + r \log_2 n$$

Divide both sides by n: $\log_2 k \leq \frac{\log_2 c}{n} + \log_2 n - \log_2 e + \frac{r}{n} \log_2 n$

The left hand side is constant, whereas the right hand side is an increasing function. Therefore, there must be some constants n_o, c and r such that the inequality is true.

Remember!

$$\boxed{\log(n!) = \Theta(n \log n)}$$

$$\begin{aligned}
\log(n!) &= \log(1 \times 2 \times 3 \times \ldots \times n-1 \times n) \\
&= \log(1) + \log(2) + \log(3) + \ldots + \log(n-1) + \log(n) \\
&\leqslant \log(n) + \log(n) + \log(n) + \ldots + \log(n) + \log(n) && n \text{ times} \\
&\leqslant n\log(n) \quad \text{for all} \quad n \geqslant 1 && \text{clearly } O(n \log n)
\end{aligned}$$

$$\begin{aligned}
\log(n!) &= \log(1 \times 2 \times \ldots \times \tfrac{n}{2} \times (\tfrac{n}{2}+1) \times \ldots \times n) \\
&= \log(1) + \log(2) + \ldots + \log(\tfrac{n}{2}) + (\log(\tfrac{n}{2}) + 1) + \ldots + \log(n) \\
&\geqslant \log(\tfrac{n}{2}) + (\log(\tfrac{n}{2}) + 1) + \ldots + \log(n) && \text{drop the first half of the sum} \\
&\geqslant \log(\tfrac{n}{2}) + \log(\tfrac{n}{2}) + \ldots + \log(\tfrac{n}{2}) && \tfrac{n}{2} \text{ times} \\
&\geqslant \tfrac{n}{2}\log(\tfrac{n}{2}) \quad \text{for all} \quad n \geqslant 1
\end{aligned}$$

a few more steps are needed to show that it is $\Omega(n \log n)$

Big-*O* Relatives

Big-Ω

Definition. Let $f(n)$ and $g(n)$ be two functions that are always positive, $f(n)$ is said to be $\Omega(g)$ if and only if :

> There are two constants c and n_o , such that
> $0 \leq c \bullet g(n) \leq f(n)$ for all $n \geq n_o$

Big-Ω

Definition. Let $f(n)$ and $g(n)$ be two functions that are always positive, $f(n)$ is said to be $\Omega(g)$ if and only if :

> There are two constants c and n_o, such that
> $0 \leq c \bullet g(n) \leq f(n)$ for all $n \geq n_o$

Less formally: If multiplying $g(n)$ by a constant makes it a lower bound for $f(n)$ after some point, then f is $\Omega(g)$.

Example # 1

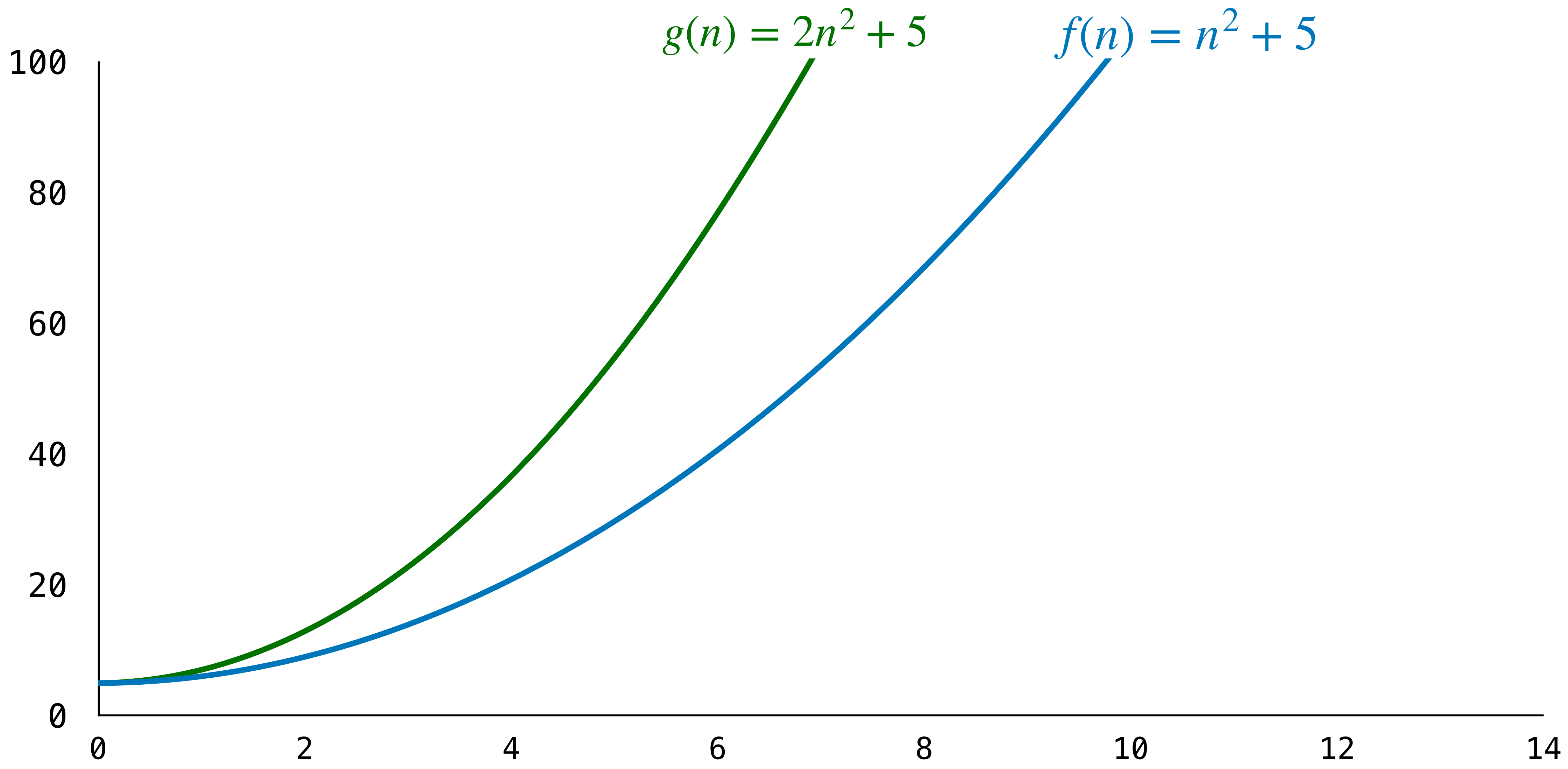
Assume $f(n) = n^2 + 5$ and $g(n) = 2n^2 + 5$.

Example # 1

Assume $f(n) = n^2 + 5$ and $g(n) = 2n^2 + 5$.

f is $\Omega(g)$ because there are c and n_o such that $0 \leq c \bullet g(n) \leq f(n)$ for all $n \geq n_o$:

If $c = \frac{1}{4}$, then $\frac{1}{4} \bullet g(n) \leq f(n)$ for all $n \geq 1$

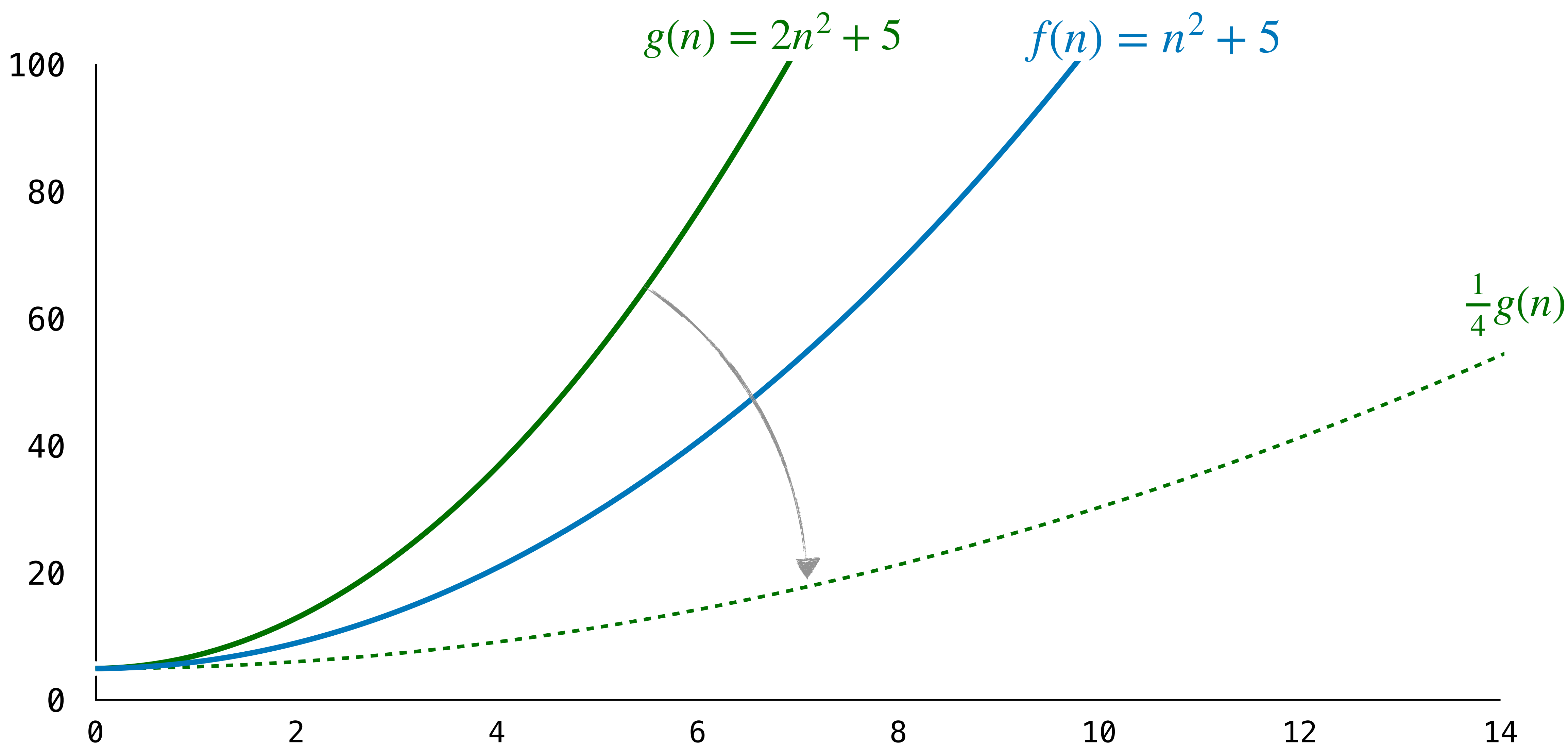


Example # 1

Assume $f(n) = n^2 + 5$ and $g(n) = 2n^2 + 5$.

f is $\Omega(g)$ because there are c and n_o such that $0 \leq c \bullet g(n) \leq f(n)$ for all $n \geq n_o$:

If $c = \frac{1}{4}$, then $\frac{1}{4} \bullet g(n) \leq f(n)$ for all $n \geq 1$



Example # 2

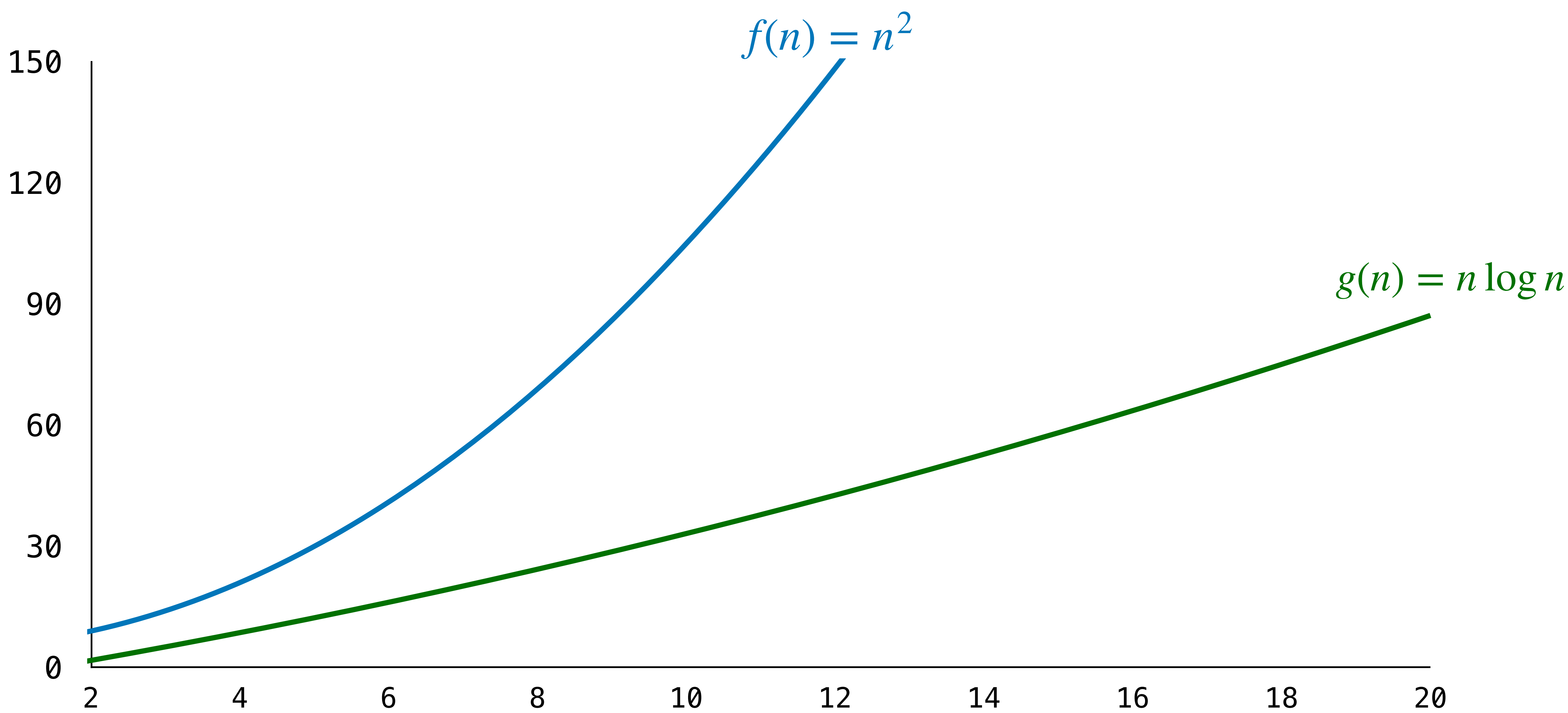
Assume $f(n) = n^2$ and $g(n) = n \log n$.

Example # 2

Assume $f(n) = n^2$ and $g(n) = n \log n$.

f is $\Omega(g)$ because there are c and n_o such that $0 \leq c \bullet g(n) \leq f(n)$ for all $n \geq n_o$:

If $c = 1$, then $g(n) \leq f(n)$ for all $n \geq 1$



Big-Θ

Definition. Let $f(n)$ and $g(n)$ be two functions that are always positive, $f(n)$ is said to be $\Theta(g)$ if and only if :

f is $O(g)$ and f is also $\Omega(g)$

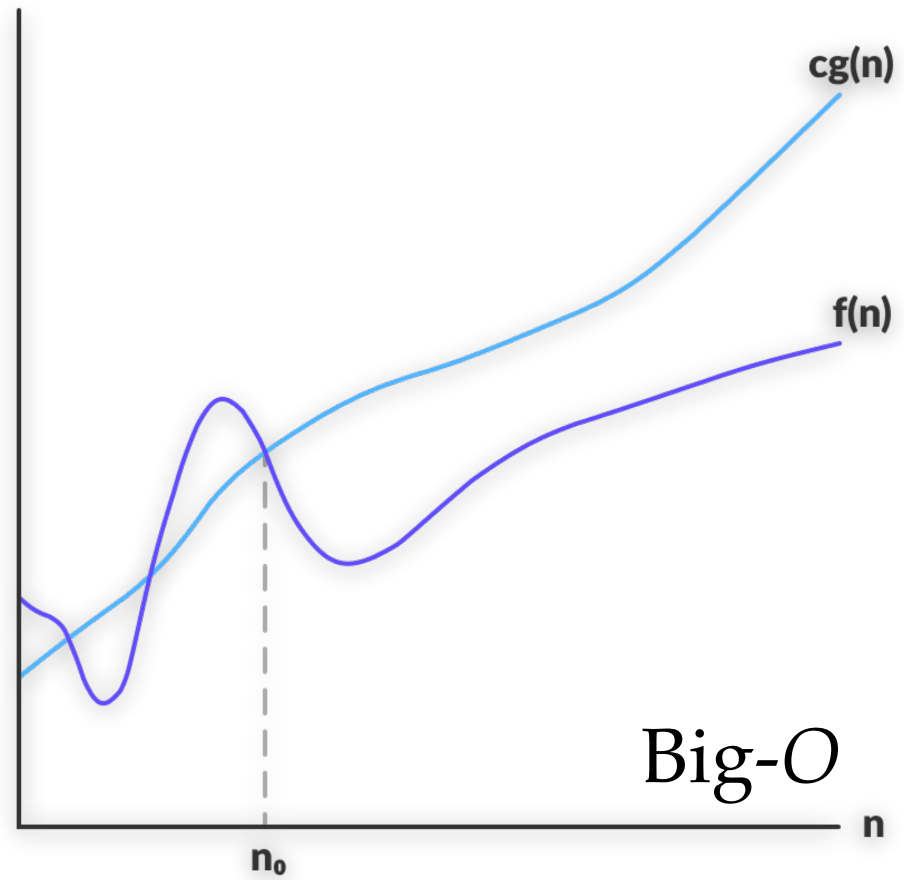
Big-Θ

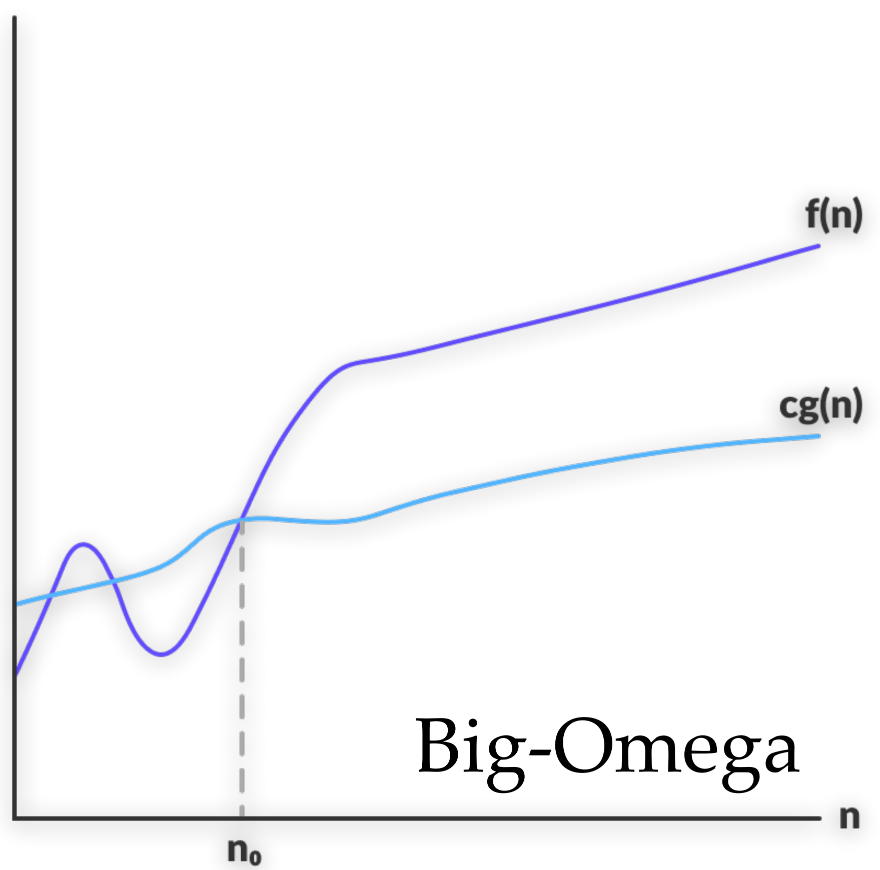
Definition. Let $f(n)$ and $g(n)$ be two functions that are always positive, $f(n)$ is said to be $\Theta(g)$ if and only if :

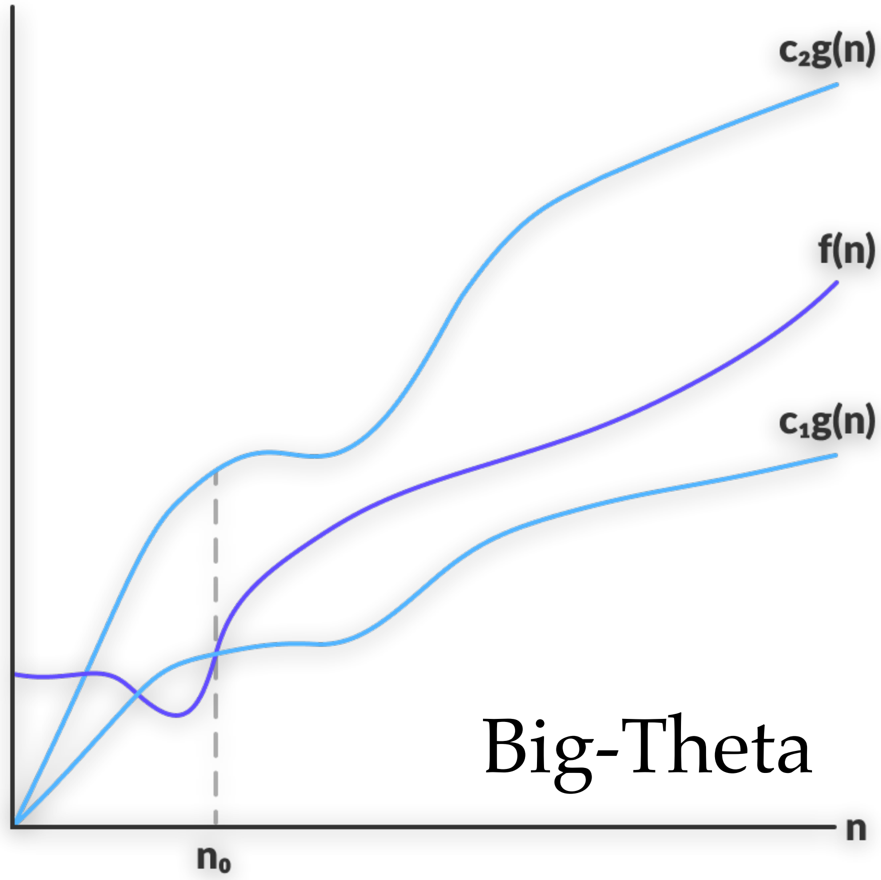
> f is $O(g)$ and f is also $\Omega(g)$

Less formally: If multiplying $g(n)$ by a constant makes it an upper bound for $f(n)$ after some point and also multiplying $g(n)$ by another constant makes it a lower bound for $f(n)$ after some point, then f is $\Theta(g)$.

Big-Θ







Exercise # 2

For each of the following function, show that f is $\Theta(n)$.

A. $f(n) = 4n + 8 \quad \text{and} \quad g(n) = n$

Exercise # 2

For each of the following function, show that f is $\Theta(n)$.

A. $f(n) = 4n + 8 \quad \text{and} \quad g(n) = n$

Solution.

We need to show that:

$$4n + 8 = O(n)$$
$$4n + 8 = \Omega(n)$$

Exercise # 2

For each of the following function, show that f is $\Theta(n)$.

A. $f(n) = 4n + 8$ and $g(n) = n$

Solution.

We need to show that:

$$4n + 8 = O(n) \longrightarrow \text{pick } c = 20 \text{ and } n_o = 1$$

$$4n + 8 = \Omega(n)$$

Exercise # 2

For each of the following function, show that f is $\Theta(n)$.

A. $f(n) = 4n + 8$ and $g(n) = n$

Solution.

We need to show that:

$$4n + 8 = O(n) \longrightarrow \text{pick } c = 20 \text{ and } n_o = 1$$

$$4n + 8 = \Omega(n) \longrightarrow \text{pick } c = 1 \text{ and } n_o = 1$$

Exercise # 2

For each of the following function, show that f is $\Theta(n)$.

A. $f(n) = 4n + 8$ and $g(n) = n$

Solution.

We need to show that:

$$4n + 8 = O(n) \longrightarrow \text{pick } c = 20 \text{ and } n_o = 1$$

$$4n + 8 = \Omega(n) \longrightarrow \text{pick } c = 1 \text{ and } n_o = 1$$

B. $f(n) = \log_2 n$ and $g(n) = \log_3 n$

Exercise # 2

For each of the following function, show that f is $\Theta(n)$.

A. $f(n) = 4n + 8$ and $g(n) = n$

Solution.

We need to show that:

$$4n + 8 = O(n) \longrightarrow \text{pick } c = 20 \text{ and } n_o = 1$$

$$4n + 8 = \Omega(n) \longrightarrow \text{pick } c = 1 \text{ and } n_o = 1$$

B. $f(n) = \log_2 n$ and $g(n) = \log_3 n$

Solution.

We need to show that:

$$\log_2 n = O(\frac{\log_2 n}{\log_2 3})$$

$$\log_2 n = \Omega(\frac{\log_2 n}{\log_2 3})$$

Exercise # 2

For each of the following function, show that f is $\Theta(n)$.

A. $f(n) = 4n + 8$ and $g(n) = n$

Solution.

We need to show that:

$$4n + 8 = O(n) \longrightarrow \text{pick } c = 20 \text{ and } n_o = 1$$

$$4n + 8 = \Omega(n) \longrightarrow \text{pick } c = 1 \text{ and } n_o = 1$$

B. $f(n) = \log_2 n$ and $g(n) = \log_3 n$

Solution.

We need to show that:

$$\log_2 n = O(\frac{\log_2 n}{\log_2 3}) \longrightarrow \text{pick } c \geq \log_2 3 \text{ and } n_o = 1$$

$$\log_2 n = \Omega(\frac{\log_2 n}{\log_2 3})$$

Exercise # 2

For each of the following function, show that f is $\Theta(n)$.

A. $f(n) = 4n + 8$ and $g(n) = n$

Solution.

We need to show that:

$4n + 8 = O(n)$ $\longrightarrow$ pick $c = 20$ and $n_o = 1$

$4n + 8 = \Omega(n)$ $\longrightarrow$ pick $c = 1$ and $n_o = 1$

B. $f(n) = \log_2 n$ and $g(n) = \log_3 n$

Solution.

We need to show that:

$\log_2 n = O(\frac{\log_2 n}{\log_2 3})$ $\longrightarrow$ pick $c \geq \log_2 3$ and $n_o = 1$

$\log_2 n = \Omega(\frac{\log_2 n}{\log_2 3})$ $\longrightarrow$ pick $c = 1$ and $n_o = 1$

More Relatives

Small-o and Small-ω

Informal Definition. f is said to be $o(g)$ if it grows strictly slower than g.

Informal Definition. f is said to be $\omega(g)$ if it grows strictly faster than g.

Small-o and Small-ω

Informal Definition. f is said to be $o(g)$ if it grows strictly slower than g.

Informal Definition. f is said to be $\omega(g)$ if it grows strictly faster than g.

Examples.

$3n^2$ vs n^2

Small-o and Small-ω

Informal Definition. f is said to be $o(g)$ if it grows strictly slower than g.

Informal Definition. f is said to be $\omega(g)$ if it grows strictly faster than g.

Examples.

$3n^2$ vs n^2

$3n^2 = \mathrm{O}(n^2)$

$3n^2 = \Omega(n^2)$

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

$3n^2 \neq \mathrm{o}(n^2)$

$3n^2 \neq \omega(n^2)$

Small-o and Small-ω

Informal Definition. f is said to be $o(g)$ if it grows strictly slower than g.

Informal Definition. f is said to be $\omega(g)$ if it grows strictly faster than g.

Examples.

$3n^2$ vs n^2	$3n^2$ vs n^3
$3n^2 = \mathrm{O}(n^2)$	
$3n^2 = \Omega(n^2)$	
$3n^2 = \Theta(n^2)$	
$3n^2 \neq \mathrm{o}(n^2)$	
$3n^2 \neq \omega(n^2)$	

Small-o and Small-ω

Informal Definition. f is said to be $o(g)$ if it grows strictly slower than g.

Informal Definition. f is said to be $\omega(g)$ if it grows strictly faster than g.

Examples.

$3n^2$ vs n^2	$3n^2$ vs n^3
$3n^2 = \mathrm{O}(n^2)$	$3n^2 = \mathrm{O}(n^3)$
$3n^2 = \Omega(n^2)$	$3n^2 \neq \Omega(n^3)$
$3n^2 = \Theta(n^2)$	$3n^2 \neq \Theta(n^3)$
$3n^2 \neq \mathrm{o}(n^2)$	$3n^2 = \mathrm{o}(n^3)$
$3n^2 \neq \omega(n^2)$	$3n^2 \neq \omega(n^3)$

Small-o and Small-ω

Informal Definition. f is said to be $o(g)$ if it grows strictly slower than g.

Informal Definition. f is said to be $\omega(g)$ if it grows strictly faster than g.

Examples.

$3n^2$ vs n^2	$3n^2$ vs n^3	$3n^3$ vs n^2
$3n^2 = \mathrm{O}(n^2)$	$3n^2 = \mathrm{O}(n^3)$	
$3n^2 = \Omega(n^2)$	$3n^2 \neq \Omega(n^3)$	
$3n^2 = \Theta(n^2)$	$3n^2 \neq \Theta(n^3)$	
$3n^2 \neq \mathrm{o}(n^2)$	$3n^2 = \mathrm{o}(n^3)$	
$3n^2 \neq \omega(n^2)$	$3n^2 \neq \omega(n^3)$	

Small-o and Small-ω

Informal Definition. f is said to be $o(g)$ if it grows strictly slower than g.

Informal Definition. f is said to be $\omega(g)$ if it grows strictly faster than g.

Examples.

$3n^2$ vs n^2	$3n^2$ vs n^3	$3n^3$ vs n^2
$3n^2 = \mathrm{O}(n^2)$	$3n^2 = \mathrm{O}(n^3)$	$3n^3 \neq \mathrm{O}(n^2)$
$3n^2 = \Omega(n^2)$	$3n^2 \neq \Omega(n^3)$	$3n^3 = \Omega(n^2)$
$3n^2 = \Theta(n^2)$	$3n^2 \neq \Theta(n^3)$	$3n^3 \neq \Theta(n^2)$
$3n^2 \neq \mathrm{o}(n^2)$	$3n^2 = \mathrm{o}(n^3)$	$3n^3 \neq \mathrm{o}(n^2)$
$3n^2 \neq \omega(n^2)$	$3n^2 \neq \omega(n^3)$	$3n^3 = \omega(n^2)$

Quiz # 2

Which of the following is true about the running time of **insertion sort**?

A. The running time is $O(n^2)$

B. The running time is $\Omega(n)$

C. The best case is $\Theta(n)$.

D. The worst case is $\Theta(n^2)$.

E. All of the above.

Quiz # 3

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

Which of the following is true?

A. $g = \Omega(f)$

B. $0 \leqslant \lim_{n \to \infty} \dfrac{f(n)}{g(n)} < \infty$

C. All of the above.

D. None of the above.

Quiz # 3

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

Which of the following is true?

A. $g = \Omega(f)$ ←

$g = \Omega(f) \iff f = O(g)$
$g = \omega(f) \iff f = o(g)$
Sketch a graph to see that it's true!

B. $0 \leqslant \lim_{n \to \infty} \frac{f(n)}{g(n)} < \infty$ ← See next slide!

C. All of the above.

D. None of the above.

Alternative Definitions

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

Alternative Definitions

if $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$ *then* $f = o(g)$ $f(n) < c \bullet g(n)$

Alternative Definitions

if $\displaystyle\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$ *then* $f = o(g)$ $f(n) < c \bullet g(n)$

if $\displaystyle\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$ *then*

Alternative Definitions

if $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$	*then*	$f = o(g)$	$f(n) < c \bullet g(n)$
if $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$	*then*	$f = \omega(g)$	$f(n) > c \bullet g(n)$

Alternative Definitions

if	$\lim\limits_{n\to\infty} \dfrac{f(n)}{g(n)} = 0$	*then*	$f = o(g)$	$f(n) < c \bullet g(n)$
if	$\lim\limits_{n\to\infty} \dfrac{f(n)}{g(n)} = \infty$	*then*	$f = \omega(g)$	$f(n) > c \bullet g(n)$
if	$0 < \lim\limits_{n\to\infty} \dfrac{f(n)}{g(n)} < \infty$	*then*		

Alternative Definitions

if	$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$	*then*	$f = o(g)$	$f(n) < c \bullet g(n)$
if	$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$	*then*	$f = \omega(g)$	$f(n) > c \bullet g(n)$
if	$0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$	*then*	$f = \Theta(g)$	not o and not ω

Alternative Definitions

if	$\lim_{n\to\infty} \dfrac{f(n)}{g(n)} = 0$	*then*	$f = o(g)$	$f(n) < c \bullet g(n)$
if	$\lim_{n\to\infty} \dfrac{f(n)}{g(n)} = \infty$	*then*	$f = \omega(g)$	$f(n) > c \bullet g(n)$
if	$0 < \lim_{n\to\infty} \dfrac{f(n)}{g(n)} < \infty$	*then*	$f = \Theta(g)$	not o and not ω
if	$0 \leqslant \lim_{n\to\infty} \dfrac{f(n)}{g(n)} < \infty$	*then*		

Alternative Definitions

if	$\lim\limits_{n\to\infty} \frac{f(n)}{g(n)} = 0$	*then*	$f = o(g)$	$f(n) < c \bullet g(n)$
if	$\lim\limits_{n\to\infty} \frac{f(n)}{g(n)} = \infty$	*then*	$f = \omega(g)$	$f(n) > c \bullet g(n)$
if	$0 < \lim\limits_{n\to\infty} \frac{f(n)}{g(n)} < \infty$	*then*	$f = \Theta(g)$	not o and not ω
if	$0 \leqslant \lim\limits_{n\to\infty} \frac{f(n)}{g(n)} < \infty$	*then*	$f = O(g)$	$f(n) \leq c \bullet g(n)$

Alternative Definitions

if	$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$	*then*	$f = o(g)$	$f(n) < c \bullet g(n)$
if	$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$	*then*	$f = \omega(g)$	$f(n) > c \bullet g(n)$
if	$0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$	*then*	$f = \Theta(g)$	not o and not ω
if	$0 \leqslant \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$	*then*	$f = O(g)$	$f(n) \leq c \bullet g(n)$
if	$0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} \leqslant \infty$	*then*		

Alternative Definitions

if	$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$	*then*	$f = o(g)$	$f(n) < c \bullet g(n)$
if	$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$	*then*	$f = \omega(g)$	$f(n) > c \bullet g(n)$
if	$0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$	*then*	$f = \Theta(g)$	not o and not ω
if	$0 \leqslant \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$	*then*	$f = O(g)$	$f(n) \leq c \bullet g(n)$
if	$0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} \leqslant \infty$	*then*	$f = \Omega(g)$	$f(n) \geq c \bullet g(n)$

for all $n \geq n_o$

Properties

- Reflexivity. f is $\Theta(f)$

Properties

- **Reflexivity.** f is $\Theta(f)$ and $O(f)$ and $\Omega(f)$ but not $o(f)$ or $\omega(f)$

Properties

- Reflexivity. f is $\Theta(f)$.
- Constants. If f is $\Theta(g)$ and $c > 0$, then $c \bullet f$ is $\Theta(g)$.

Properties

- Reflexivity. f is $\Theta(f)$.

- Constants. If f is $\Theta(g)$ and $c > 0$, then $c \bullet f$ is $\Theta(g)$.

  Example: $4n^2 + 5$ is $\Theta(n^2)$ and $4 \times (4n^2 + 5)$ is also $\Theta(n^2)$.

Properties

- Reflexivity. f is $\Theta(f)$.

- Constants. If f is $\Theta(g)$ and $c > 0$, then $c \bullet f$ is $\Theta(g)$.

  Example: $4n^2 + 5$ is $\Theta(n^2)$ and $4 \times (4n^2 + 5)$ is also $\Theta(n^2)$.

  Similarly: If f is $O(g)$ and $c > 0$, then $c \bullet f$ is $O(g)$.

  If f is $\Omega(g)$ and $c > 0$, then $c \bullet f$ is $\Omega(g)$.

  If f is $o(g)$ and $c > 0$, then $c \bullet f$ is $o(g)$.

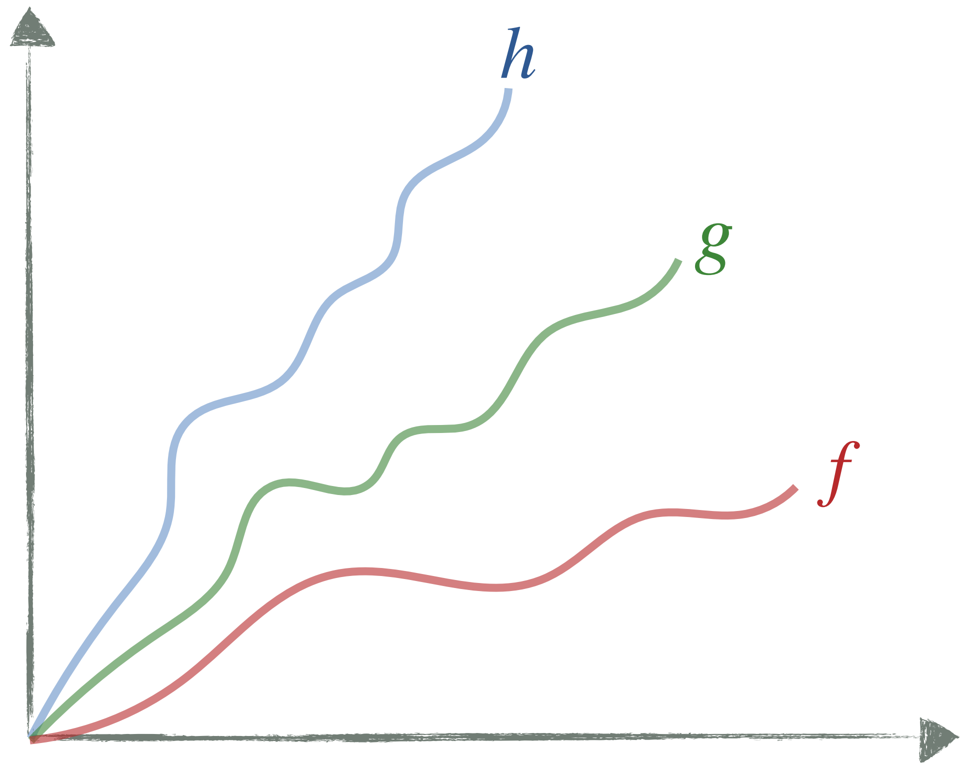
  If f is $\omega(g)$ and $c > 0$, then $c \bullet f$ is $\omega(g)$.

Properties

- Reflexivity. f is $\Theta(f)$.
- Constants. If f is $\Theta(g)$ and $c > 0$, then $c \bullet f$ is $\Theta(g)$.
- Transitivity. If f is $O(g)$ and g is $O(h)$ then f is $O(h)$.

Properties

- Reflexivity. f is $\Theta(f)$.
- Constants. If f is $\Theta(g)$ and $c > 0$, then $c \bullet f$ is $\Theta(g)$.
- Transitivity. If f is $O(g)$ and g is $O(h)$ then f is $O(h)$.



h is an upper bound for both g and f

Properties

- Reflexivity. f is $\Theta(f)$.

- Constants. If f is $\Theta(g)$ and $c > 0$, then $c \bullet f$ is $\Theta(g)$.

- Transitivity. If f is $O(g)$ and g is $O(h)$ then f is $O(h)$.

  Similarly: If f is $\Theta(g)$ and g is $\Theta(h)$ then f is $\Theta(h)$.

  If f is $\Omega(g)$ and g is $\Omega(h)$ then f is $\Omega(g)$.

  If f is $o(g)$ and g is $o(h)$ then f is $o(h)$.

  If f is $\omega(g)$ and g is $\omega(h)$ then f is $\omega(h)$.

Properties

- Reflexivity. f is $\Theta(f)$.
- Constants. If f is $\Theta(g)$ and $c > 0$, then $c \bullet f$ is $\Theta(g)$.
- Transitivity. If f is $\Theta(g)$ and g is $\Theta(h)$ then f is $\Theta(h)$.
- Sums. If f_1 is $\Theta(g_1)$ and f_2 is $\Theta(g_2)$, then $f_1 + f_2$ is $\Theta(\max\{g_1, g_2\})$.

  Example: If $f_1(n)$ is $\Theta(n^2)$ and $f_2(n)$ is $\Theta(n^3)$ then $f_1 + f_2$ is $\Theta(n^3)$.

Properties

- Reflexivity. f is $\Theta(f)$.
- Constants. If f is $\Theta(g)$ and $c > 0$, then $c \bullet f$ is $\Theta(g)$.
- Transitivity. If f is $\Theta(g)$ and g is $\Theta(h)$ then f is $\Theta(h)$.
- Sums. If f_1 is $\Theta(g_1)$ and f_2 is $\Theta(g_2)$, then $f_1 + f_2$ is $\Theta(\max\{g_1, g_2\})$.
  Example: If $f_1(n)$ is $\Theta(n^2)$ and $f_2(n)$ is $\Theta(n^3)$ then $f_1 + f_2$ is $\Theta(n^3)$.

! $O(g(n))$ is a set of functions, but computer scientists often *abuse* the notation by writing $f(n) = O(g(n))$ instead of $f(n) \in O(g(n))$.

Caution!

Notes of Caution

- Same worst case. Two algorithms with the same worst case order of growth of the running time are not necessarily equally fast in practice!

Notes of Caution

- Same worst case. Two algorithms with the same worst case order of growth of the running time are not necessarily equally fast in practice!

  Example: The worst case of Timsort and Heapsort is $\Theta(n \log n)$, but Timsort is faster in practice.

Notes of Caution

- Same worst case. Two algorithms with the same worst case order of growth of the running time are not necessarily equally fast in practice!

  Example: The worst case of Timsort and Heapsort is $\Theta(n \log n)$, but Timsort is faster in practice.

- Worse worst case. An algorithm with a worse order of growth of the running time in the worst case is not necessarily slower in practice!

Notes of Caution

- Same worst case. Two algorithms with the same worst case order of growth of the running time are not necessarily equally fast in practice!

  Example: The worst case of Timsort and Heapsort is $\Theta(n \log n)$, but Timsort is faster in practice.

- Worse worst case. An algorithm with a worse order of growth of the running time in the worst case is not necessarily slower in practice!

  Example: Heapsort has a worst case of $\Theta(n \log n)$ and Quicksort has a worst case of $\Theta(n^2)$, but Quicksort is faster in practice!

Notes of Caution

- Same worst case. Two algorithms with the same worst case order of growth of the running time are not necessarily equally fast in practice!

  Example: The worst case of Timsort and Heapsort is $\Theta(n \log n)$, but Timsort is faster in practice.

- Worse worst case. An algorithm with a worse order of growth of the running time in the worst case is not necessarily slower in practice!

  Example: Heapsort has a worst case of $\Theta(n \log n)$ and Quicksort has a worst case of $\Theta(n^2)$, but Quicksort is faster in practice!

- Which is better, a $\Theta(2^{\sqrt{n}})$ algorithm or a $\Theta(n^{20})$ algorithm?
  Although $\Theta(2^{\sqrt{n}})$ grows faster, it is performs less operations for $n \lessapprox 112000$.

Notes of Caution

- **Same worst case.** Two algorithms with the same worst case order of growth of the running time are not necessarily equally fast in practice!

  Example: The worst case of Timsort and Heapsort is $\Theta(n \log n)$, but Timsort is faster in practice.

- **Worse worst case.** An algorithm with a worse order of growth of the running time in the worst case is not necessarily slower in practice!

  Example: Heapsort has a worst case of $\Theta(n \log n)$ and Quicksort has a worst case of $\Theta(n^2)$, but Quicksort is faster in practice!

- Which is better, a $\Theta(2^{\sqrt{n}})$ algorithm or a $\Theta(n^{20})$ algorithm?
  Although $\Theta(2^{\sqrt{n}})$ grows faster, it is performs less operations for $n \lessapprox 112000$.

! To compare the **actual running time** of algorithms, other factors need to be taken into account (e.g. typical input sizes, likelihood of worst case, constant factors, lower order terms).