Big-Omega & Big-Theta

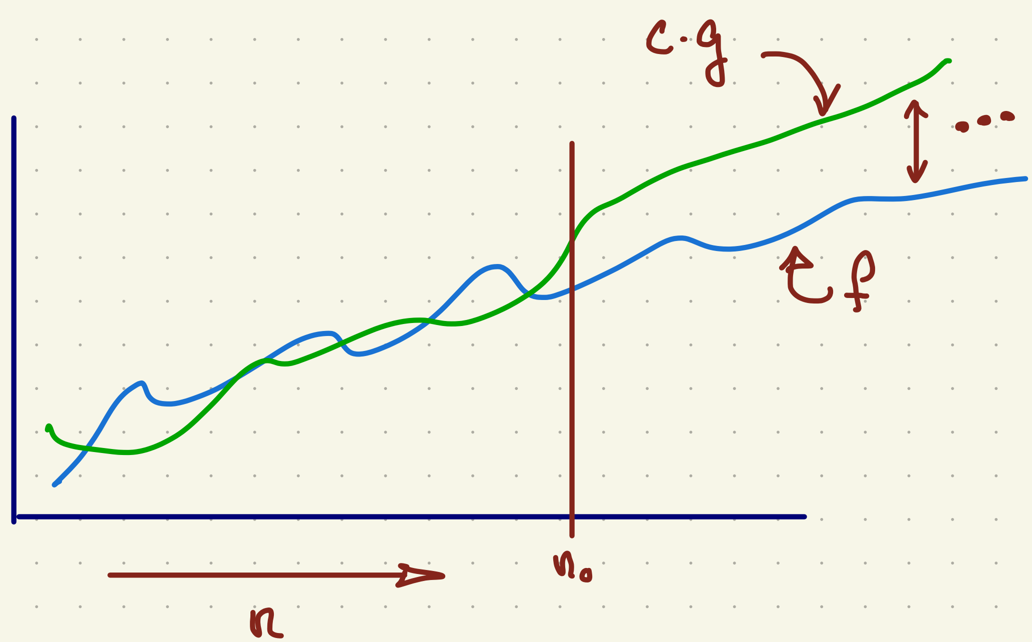
Big-Oh Gives Upper Bounds

For f, g functions $\mathbb{N} \to \mathbb{R}^+$,

$f(n) = O(g(n))$ means there are $c, n_0 > 0$ s.t.

$$\forall n > n_0 \quad f(n) \leq c \cdot g(n)$$

ie. f is asymptotically bounded from above by g



or. f grows asymptotically no faster than g.

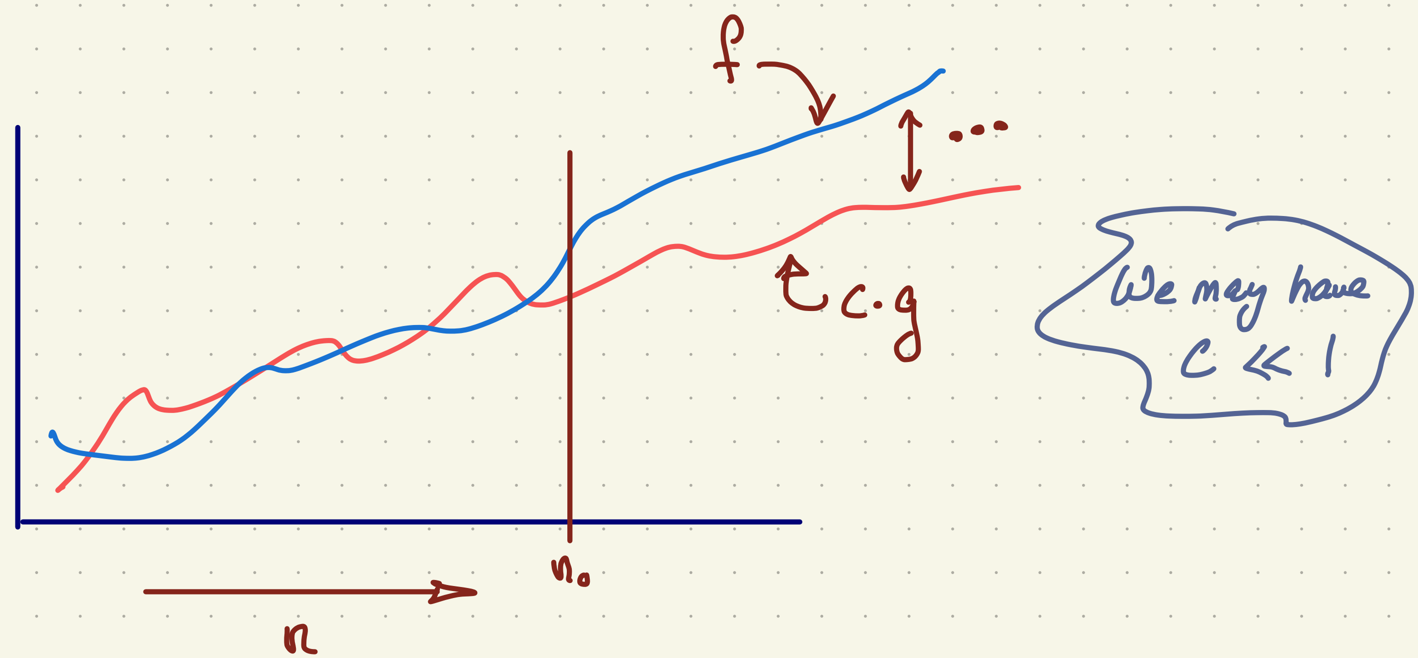
Big-Omega Gives Analogous Lower Bounds

For f, g functions $\mathbb{N} \to \mathbb{R}^+$,

$f(n) = \Omega(g(n))$ means there are $c, n_0 > 0$ s.t.

$$\forall n > n_0 \quad f(n) \geq c \cdot g(n)$$

ie f is asymptotically bounded from below by g



or f grows asymptotically at least as fast as g.

Big-Oh & Big-Omega are Duals

<u>Fact</u>: $f(n) = \Omega(g(n)) \iff g(n) = O(f(n))$

<u>Pf</u>: $f(n) = \Omega(g(n)) \iff \exists n_0, c' > 0$ s.t. $n > n_0 \Rightarrow f(n) \geq c' \cdot g(n)$

$\iff \exists n_0, c' > 0$ s.t. $n > n_0 \Rightarrow c' \cdot g(n) \leq f(n)$

$\iff \exists n_0, c > 0$ s.t. $n > n_0 \Rightarrow g(n) \leq c \cdot f(n)$ // letting $c = \frac{1}{c'}$

$\iff g(n) = O(f(n))$ $\boxed{}$

So: f grows at least as fast as g

$\iff g$ grows at most as fast as f.

Examples: Worst-case times

Operation	$O(1)$	$\Omega(1)$	$O(\log n)$	$\Omega(\log n)$	$O(n)$	$\Omega(n)$	$O(n \log n)$	$\Omega(n \log n)$
· stack push/pop	✓	✓	✓	✗	✓	✗	✓	✗
· enqueue/dequeue	✓	✓	✓	✗	✓	✗	✓	✗
· heap insert or extract min	✗	✓	✓	✓	✓	✗	✓	✗
· AVL-tree find, insert, remove	✗	✓	✓	✓	✓	✗	✓	✗
· make_heap	✗	✓	✗	✓	✓	✓	✓	✗
· BST find, insert remove	✗	✓	✗	✓	✓	✓	✓	✗
· Sorting	✗	✓	✗	✓	✗	✓	✓	?

Examples: Worst-case times

Operation	O(1)	Ω(1)	O(log n)	Ω(log n)	O(n)	Ω(n)	O(n log n)	Ω(n log n)
stack push/pop	✓	✓	✓	✗	✓	✗	✓	✗
enqueue/dequeue	✓	✓	✓	✗	✓	✗	✓	✗
heap insert or extract min	✗	✓	✓	✓	✓	✗	✓	✗
AVL-tree find, insert, remove	✗	✓	✓	✓	✓	✗	✓	✗
make_heap	✗	✓	✗	✓	✓	✓	✓	✗
BST find, insert remove	✗	✓	✗	✓	✓	✓	✓	✗
Sorting	✗	✓	✗	✓	✗	✓	✓	?

Examples: Worst-case times

Operation	$O(1)$	$\Omega(1)$	$O(\log n)$	$\Omega(\log n)$	$O(n)$	$\Omega(n)$	$O(n \log n)$	$\Omega(n \log n)$
· stack push/pop	✓	✓	✓	✗	✓	✗	✓	✗
· enqueue/dequeue	✓	✓	✓	✗	✓	✗	✓	✗
· heap insert or extract min	✗	✓	✓	✓	✓	✗	✓	✗
· AVL-tree find, insert, remove	✗	✓	✓	✓	✓	✗	✓	✗
· make_heap	✗	✓	✗	✓	✓	✓	✓	✗
· BST find, insert remove	✗	✓	✗	✓	✓	✓	✓	✗
· Sorting	✗	✓	✗	✓	✗	✓	✓	?

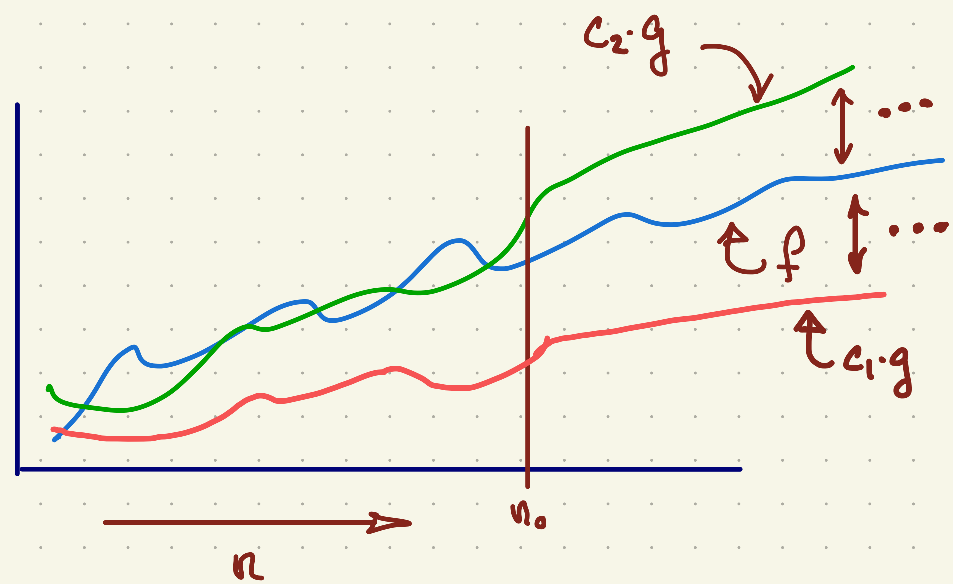
Big-Theta Expresses "Tight Bounds"

For f, g functions $\mathbb{N} \to \mathbb{R}^+$,

$f(n) = \Theta(g(n))$ means there are $c_1, c_2, n_0 > 0$ s.t.

$$n > n_0 \Rightarrow c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$$

ie. f is asymptotically bounded from above and below by g



or. f grows asymptotically the same as g.

"Grows the same as" is symmetric

Fact: $f(n) = \Theta(g(n)) \iff g(n) = \Theta(f(n))$

i.e, f grows the same as g $\iff$ g grows the same as f.

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

$\iff \exists c1, c2, n_0 > 0$ s.t $\forall n > n_0$ $c1 \cdot g(n) \leq f(n) \leq c2 \cdot g(n)$

$\iff \exists c_1, c2, n_0 > 0$ s.t. $\forall n > n_0$ $\frac{1}{c2} f(n) \leq g(n) \leq \frac{1}{c_1} f(n)$

$\iff \exists c_3, c_4, n_0 > 0$ s.t. $\forall n > n_0$ $c_3 f(n) \leq g(n) \leq c_4 f(n)$

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

End