Algorithms

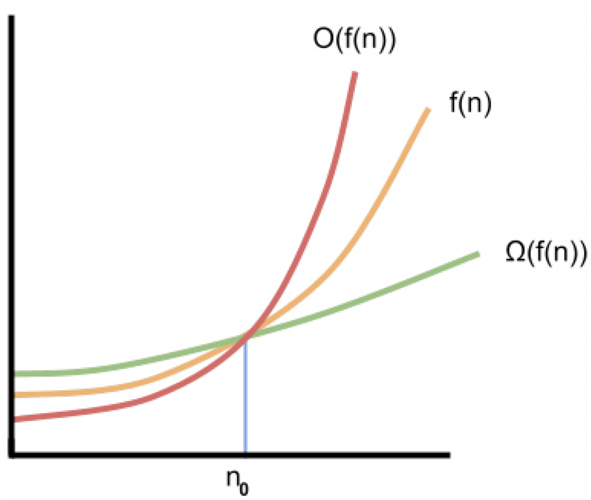
---

1 Theta Θ

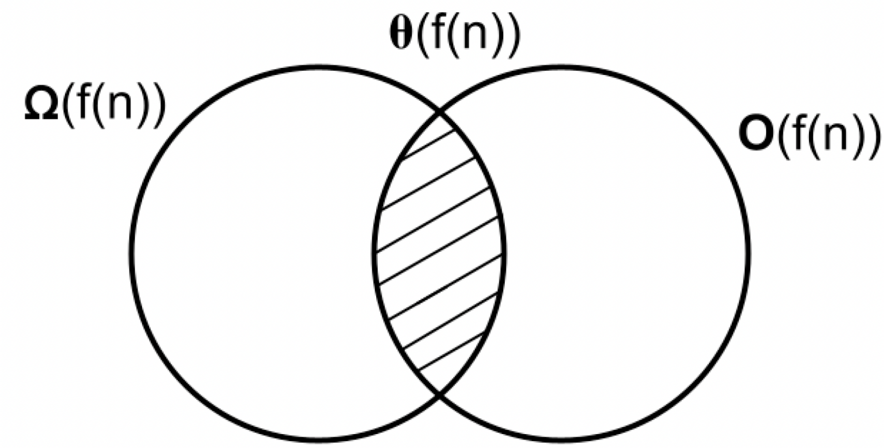
Defintion 1. For a given complexity function, $f(n)$, $\theta(f(n))$ is the set of complexity functions $g(n)$ for which there exists 2 positive real constants, C_1 and C_2, and a non-negative integer n_0 such that for all $n \geq 0$

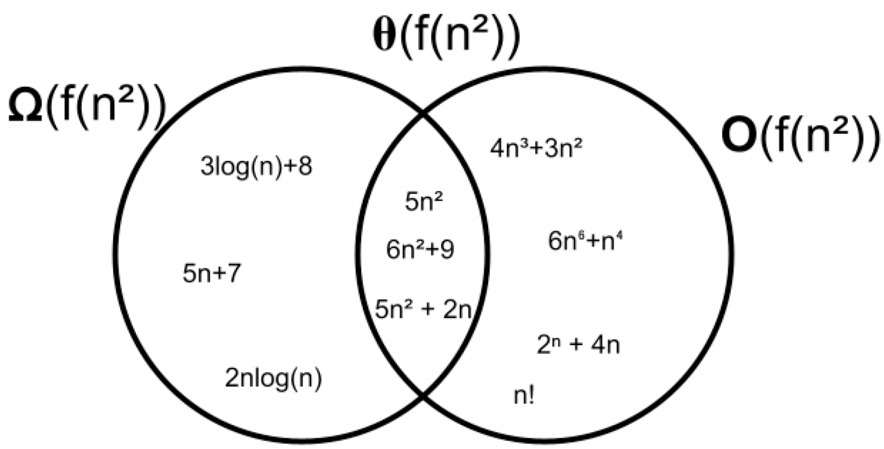
$$C_1 \cdot f(n) \leq g(n) \leq C_2 \cdot f(n)$$

Theta is the intersection of Big-O and Big-Ω: it is an upper and lower bound on $f(n)$. There are two constants C_1 and C_2 associated with theta rather than just one, as with Big-O and Big-Ω. C_1 makes the lower bound and C_2 makes the upper.



To show $g(n) \in \Theta(f(n))$, one must show that $g(n) \in O(f(n))$ AND $g(n) \in \Omega(f(n))$. In order to show that $g(n) \notin \Theta(f(n))$, one can show $g(n) \notin O(f(n))$ OR $g(n) \notin \Omega(f(n))$. This can be visualized easily using a set diagram.





Example 1.1. Show that $5n^2 \in \Theta(n^2)$.

First show $5n^2 \in O(n^2)$:

$$5n^2 \leq C \cdot n^2$$

Letting $C = 5$ will make both sides of the equation equal.

$$5^2 \leq 5n^2$$

No matter the value of n, the sides will be the same, so we will make the arbitrary choice to let $n_0 = 0$.

Now, show that $5n^2 \in \Omega(n^2)$.

$$5n^2 \geq C \cdot n^2$$

Once again, letting $C = 5$ will make both sides the same, so we can let $n_0 = 0$ again.

Since $5n^2 \in O(n^2)$ and $5n^2 \in \Omega(n^2)$, $5n^2 \in \Theta(n^2)$. For $n \geq 0$:

$$5n^2 \leq 5n^2 \leq 5n^2$$

Note that C_1 and C_2 do not have to share the same value as they do in this case. We could let $C = 6$ for $5n^2 \in O(n^2)$ and $C = 4$ for $5n^2 \in \Omega(n^2)$, for example (n_0 remains 0) and still have a valid inequality:

$$4n^2 \leq 5n^2 \leq 6n^2$$

Example 1.2. Show $n \notin \Theta(n^2)$.

To show $n \notin \Theta(n^2)$, we can show $n \notin \Omega(n^2)$. In order to do so, we can perform a proof by contradiction. To do a proof by contradiction:

Assume the claim $n \notin \Omega(n^2)$ is false.

If it is false, then $n \in \Omega(n^2)$.

By the definition of Ω, there is a positive real constant C and a non-negative integer n_0 such that for all $n \geq n_0$

$$n \geq C \cdot n^2$$

Divide both sides by Cn:

$$\frac{1}{C} \geq n$$

As n grows unbounded, this will become false, no matter what value we give to C. Specifically, it will become false when $n > \frac{1}{C}$.

This contradicts the statement $n \in \Omega(n^2)$, therefore

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

and by the definition of Θ

$$n \notin \Theta(n^2)$$

2 Little-o

Defintion 2. For a given complexity function, $f(n)$, $o(f(n))$ is the set of complexity functions $g(n)$ that satisfies the following: For every positive real constant C, there exists a non-negative integer n_0 such that for all $n \geq n_0$

$$g(n) \leq C \cdot f(n)$$

The distinction between little-o and the previous definitions for Big-O, Big-Ω, and Θ is that there must be an n_0 for all possible values of C, rather than requiring that there exists at least one valid set of values. Obviously, we can't find an n_0 for each C by hand; instead, define the value of n using C as in the following example.

Example 2.1. Show $n \in o(n^2)$. Let $C > 0$ be given.

Solve for n.

$$n \leq C \cdot n^2$$

Divide both sides by Cn.

$$\frac{1}{C} \leq n$$

So, $n \geq \frac{1}{C}$. We can define n_0 as a function of C. In order to make sure n is an integer, we can use either a floor or ceiling function on $\frac{1}{C}$ to round it either down or up, respectively. Either works here; we will make the arbitrary choice to use floor.

$$n_0 = \lfloor \frac{1}{C} \rfloor$$

To prove $g(n) \notin o(f(n))$, we can do a proof by contradiction following a similar logic as with theta.

Example 2.2. Show $n \notin o(5n)$. To prove by contradiction, suppose $n \in o(5n)$.

By the definition of Little-o, there must exist a non-negative integer n_0 for every value of C.

$$n \leq C \cdot 5n$$

Let $C = \frac{1}{6}$:

$$n \leq \frac{1}{6} \cdot 5n$$

$$n \leq \frac{5}{6} n$$

$$1 \leq \frac{5}{6}$$

This is a contradiction. Therefore, $n \notin o(5n)$.

3 Little-omega ω

Defintion 3. For a given complexity function, $f(n)$, $\omega(f(n))$ is the set of complexity functions $g(n)$ that satisfies the following: For every positive real constant C, there exists a non-negative integer n_0 such that for all $n \geq n_0$

$$g(n) \geq C \cdot f(n)$$

Similar to Little-o, this means that a function $g(n)$ is considered $\omega(f(n))$ if an initial value n_0 can be found that satisfies the equation for every positive real constant C. Proving a statement true or false is done in the same way as Little-o.

Example 3.1. Show $2n^2 \in \omega(n)$. Let $C > 0$ be given.

Solve for n.

$$2n^2 \geq C \cdot n$$

Divide both sides by 2n:

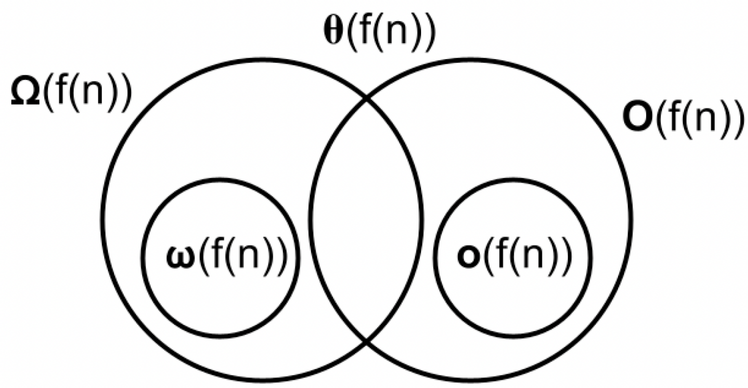
$$n \geq \frac{C}{2}$$

As with the example with little-o, we will define n_0 as a function of C. This time, we'll choose the ceiling function, but as before, either would work.

$$n_0 = \left\lceil \frac{C}{2} \right\rceil$$

4 Final Notes

The introduction of Little-o and Little-omega allows our set diagram to display a more complete picture.



The complexity functions can be thought of analogous to the relationships between x and y:

$x < y$	$g(n) \in o(f(n))$
$x \leq y$	$g(n) \in O(f(n))$
$x = y$	$g(n) \in \Theta(f(n))$
$x \geq y$	$g(n) \in \Omega(f(n))$
$x < y$	$g(n) \in \omega(f(n))$

(If this doesn't make sense to you, ignore it; it's just another way to think about the complexity functions that some may find more intuitive.)