Asymptotic Notation and Growth of Functions (Sections 3.1 and 3.2)

Running time depends on the size of the input. In the previous example that was n.

- Larger array takes longer to sort.
- $T(n)$: the time taken on an input of size n.
- *Asymptotic analysis* considers the growth of $T(n)$ as n goes to infinity.

Intuition about asymptotic growth:

- n dominates constant factors:

- higher-order terms dominate lower-order terms:

Graph of $5n^2$ and $100n$:

We use O, Ω, and Θ to specify the runtimes of algorithms. In particular, these specify sets of runtime functions.

Big-Oh:
$f(n) \in O(g(n)) \Leftrightarrow \exists_{c>0} \exists_{n_0>0} \forall_{n \geq n_0} : f(n) \leq c \cdot g(n)$

Big-Omega:
$f(n) \in \Omega(g(n)) \Leftrightarrow \exists_{c>0} \exists_{n_0>0} \forall_{n \geq n_0} : f(n) \geq c \cdot g(n)$

Theta:
$f(n) \in \Theta(g(n)) \Leftrightarrow f(n) \in O(g(n)) \wedge f(n) \in \Omega(g(n))$

(Alternative definition)
$f(n) \in \Theta(g(n)) \Leftrightarrow$
$\exists_{c_1>0} \exists_{c_2>0} \exists_{n_0>0} \forall_{n \geq n_0} : c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$

Little-Oh:
$f(n) \in o(g(n)) \Leftrightarrow \forall_{c>0} \exists_{n_0>0} \forall_{n \geq n_0} : f(n) < c \cdot g(n)$

Anything slower or equal in growth of n2 is in this set

These represent sets of functions. For example:

$$O(n^2) = \{ n^2, 10n^2, 1000n, 13, \log n, \ldots, \}$$

Example 1: goal

$100n \in O(5n^2)$ //Find a c to satisfy this statement

$\Rightarrow 100n \leq c \cdot 5n^2$

$\therefore$ goal is to go from $100n$ until we get to RHS

$\Rightarrow 100n \leq \ldots \leq \ldots \leq \ldots \leq c \cdot 5n^2$

$\Rightarrow 100n \leq 100n \cdot n = 100n^2 = 20 \cdot 5n^2$; $c = 20$

$100 = 20 \cdot 5$

$\Rightarrow$ Therefore, $c = 20$, and $n_0 = 1$

Example 2: $\therefore$ goal is to have LHS become RHS

$n^2 \in O(4n^2)$

$\Rightarrow n^2 \leq c \cdot 4n^2$

$\Rightarrow n^2 \leq \ldots \leq \ldots \leq c \cdot 4n^2$

$\Rightarrow n^2 \leq 4 \cdot n^2$

$\Rightarrow$ therefore $c = 1$, and $n_0 = 1$

Example 3:

$4n^2 \in O(n^2)$ = "$4n^2$ is upper bounded by n^2"

$\Rightarrow 4n^2 \leq c \cdot n^2$

$\therefore$ easy one as 4 is already our constant

$\Rightarrow$ Therefore, $c = 4$ & $n_0 = 1$

Example 4: //What can we do to make LHS bigger so it looks like n^3

$3n^3 + 3n^2 - 3n \in O(n^3)$ for some c & N_0

$\Rightarrow 3n^3 + 3n^2 - \underbrace{3n}_{\text{dont want}} \leq c \cdot n^3$

$\Rightarrow 3n^3 + 3n^2 - \cancel{3n} + \cancel{3n} \Rightarrow 3n^3 + \underbrace{3n^2}_{\text{cant subtract here as that would make LHS smaller}} \leq c \cdot n^3$

$\Rightarrow 3n^3 + \underbrace{3n^2}_{\text{multi. by another } n} \leq c \cdot n^3$

$\Rightarrow 3n^3 + 3n^2 \cdot \overset{n \geq 1}{n} = 3n^3 + 3n^3 = 6n^3$

$\Rightarrow 6n^3 \leq c \cdot n^3$

$\Rightarrow$ Therefore, $c = 6$ & $n_0 = 1$ $\square$

Alternative solution:

$\Rightarrow 3n^3 + 3n^2 \leq c \cdot n^3$

replace 3 w/ n

$\Rightarrow 3n^3 + n \cdot n^2 \Rightarrow 3n^3 + n^3 \leq c \cdot n^3$

$\Rightarrow 4n^3 \leq c \cdot n^3$

$\Rightarrow$ Therefore, $c = 4$, $n_0 = 3$ $\square$

* Because we replaced 3 w/ n. This tells us that $n \geq 3$
$\Rightarrow n_0 = 3$

* This is useful when we have (-) numbers Or, are doing an Ω proof.

Example 5:

$n \in O(n^4)$

$\Rightarrow n \leq c \cdot n^4$

$\Rightarrow n \leq n \cdot n^3 = n^4$

↳ Can multi. by any # of n. In this case n^3

Therefore, true for $c = 1$, $n_0 = 1$

Example 6:

$4n^3 \in \Omega(n)$ = "$4n^3$ is lower bounded by n"

∴ Omega wants to make f(n) smaller which is the opposite of big O

$\Rightarrow 4n^3 \geq c \cdot n$

$\Rightarrow 4n^3 \geq \ldots \geq \ldots \geq c \cdot n$

$\Rightarrow 4n^3 = 4n \cdot \underbrace{n \cdot n}_{\text{replace w/ 1's}} \geq 4n \cdot 1 \cdot 1 = 4n$

$\Rightarrow$ Therefore, true when $c = 4$ & $n_0 = 1$

Example 7: $3n^3 - 3n^2 - 3n \in \Omega(n^3)$

$\therefore$ assume p is True to prove q

$p \rightarrow q$

Oh-Omega Lemma: $f(n) \in O(g(n)) \Leftrightarrow g(n) \in \Omega(f(n))$

Proof: $f(n) \in O(g(n)) \Rightarrow g(n) \in \Omega(f(n))$

$\Rightarrow$ Assume: $f(n) \in O(g(n)) \Leftrightarrow f(n) \leq c_1 \cdot g(n) \; \forall n \geq n_1$

Prove: $g(n) \in \Omega(f(n)) \Leftrightarrow g(n) \geq c_2 \cdot f(n) \; \forall n \geq n_2$

$\Rightarrow f(n) \leq c_1 g(n) \Rightarrow \dfrac{\cancel{c_1} g(n)}{\cancel{c_1}} \geq \dfrac{f(n)}{c_1}$

$\Rightarrow g(n) \geq \frac{1}{c_1} f(n)$; $c_1 > 0$

$\Rightarrow$ let $\frac{1}{c_1} = c_2$

$\Rightarrow$ Therefore, $g(n) \geq c_2 f(n)$ $\square$

// To fully complete proof, we must show in reverse order

// It can be said every Big O statement is a Big Ω statement (vice versa)

Example 7:
$4n^2 \in \Theta(n^2)$ = " $4n^2$ is tightly bounded by n^2 "

$\Rightarrow \underbrace{4n^2 \in O(n^2)}_{\text{from example 3: } c = 4,\ n_0 = 1}$ & $\underbrace{4n^2 \in \Omega(n^2)}_{\text{★ trick}}$

$\Rightarrow$ you can say that $\boxed{4n^2 \in \Omega(n^2) \Leftrightarrow n^2 \in O(4n^2)}$ from theorem above

$\Rightarrow$ Thus from example 2
$c = 1$, $n_0 = 1$

$\Rightarrow$ Therefore true as we have proved O and Ω

Properties of o:

$$
\begin{aligned}
&(1)\ \ f(n) \in o(g(n)) \Rightarrow f(n) \in O(g(n)) \\
&(2)\ \ f(n) \in o(g(n)) \Leftrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0
\end{aligned}
$$

Limit theorem: If $\lim_{n\to\infty} \frac{f(n)}{g(n)} =$

$$
\begin{aligned}
&(1)\ \ 0 \Rightarrow f(n) \in o(g(n)) \\
&(2)\ \ c \neq 0 \Rightarrow f(n) \in \Theta(g(n)) \\
&(3)\ \ \infty \Rightarrow g(n) \in o(f(n))
\end{aligned}
$$

Example 8:
$n \in o(n^2)$

Both going to ∞, which are growing faster?

$$\Rightarrow \lim_{n\to\infty} \frac{\cancel{n}}{\cancel{n^2}} = \lim_{n\to\infty} \frac{1}{n} = \frac{1}{\infty} = 0$$

$$\Rightarrow \lim_{n\to\infty} \frac{1}{n} = 0, \text{ thus } n \in o(n^2)$$

Example 9:
$n \notin o(n)$

$$\Rightarrow \lim_{n\to\infty} \frac{n}{n} = \lim_{n\to\infty} 1 = 1 \quad // \text{ constant } != 0$$

$\Rightarrow$ Therefore, $n \in \Theta(n)$ & $n \notin o(n)$

$\therefore \log_2 n \in \Theta(\log_3 n)$ & $\log_2 n \in \Theta(\log_{1000} n)$

=> regardless of base, these are asymptotically the same, bases will not effect the runtime

Example 10:

$\log_b n \in \Theta(\log_c n)$

Use Log-rule:

$\log_b n = \frac{\log_c n}{\log_c b}$

$$\Rightarrow \lim_{n\to\infty} \frac{\log_b n}{\log_c n} = \lim_{n\to\infty} \frac{\frac{\log_c n}{\log_c b}}{\frac{\log_c n}{1}} = \lim_{n\to\infty} \frac{\cancel{\log_c n}}{\log_c b} \cdot \frac{1}{\cancel{\log_c n}}$$

multi. by reciprocal

$$= \lim_{n\to\infty} \frac{1}{\log_c b}$$

// c & b are constants

$$\Rightarrow \lim_{n\to\infty} \frac{1}{\log_c b} = \underbrace{\frac{1}{\log_c b}}_{\text{constant } != 0}$$

=> Therefore, $\log_b n \in \Theta(\log_c n)$ $\square$

// use lim theorem

Example 11:

$\underbrace{2^n}_{\underbrace{2\,2\,2\,2\ldots 2\,2}_{n \text{ copies}}} \notin \Theta(\underbrace{3^n}_{\underbrace{3\,3\,3\ldots 3\,3}_{n \text{ copies}}})$

$$\Rightarrow \lim_{n\to\infty} \frac{2^n}{3^n} = \lim_{n\to\infty} \left(\frac{2}{3}\right)^n = 0$$

// because of fraction multi.

$$\therefore \frac{2}{3} \cdot \frac{2}{3} \cdot \frac{2}{3} \cdot \frac{2}{3} \ldots \longrightarrow \infty$$

$$\Rightarrow 2^n \in o(3^n) \;\&\; 2^n \notin \Theta(3^n)$$

Run time of code snippets:

```
j = 1;  — 1 time
while j <= n do  —(n+1) time
    j++;  — n  time
end while
```

$\therefore j = \underbrace{1, 2, 3, 4, \ldots, n, \underbrace{n+1}_{\text{fail case}}}_{n+1}$

$\Rightarrow T(n) = 2n+2$

$\Rightarrow \in \Theta(n)$

```
j = 10;  ~  1  time
while j <= n do
    j++;
end while
```

$\Rightarrow \Theta(n)$

Run time of code snippets (cont.): e/a time j++ we run inner loop n times

nested loop

```
j = 1;
while j <= n do
    i = 1;
    while i <= n do
        i++;
    end while
    j++;
end while
```

— $O(n)$

$O(n)$ { $j = 1$, $i = 1, 2, 3, 4, \dots n$
$j = 2$, $\underbrace{i = 1, 2, 3, 4 \dots n}_{O(n)}$
$\vdots$
$j = n$ }

$\Rightarrow O(n^2)$

```
j = 1;
while j <= n do
    j = j * 2;
end while
```

$\Rightarrow 2^x = n$ $\Rightarrow$ breakout of loop

$\Rightarrow \cancel{\log_2}(\cancel{2^x}) = \log_2(n)$

$\Rightarrow x = \log_2 n$ $\Rightarrow O(\log n)$

$2^0 \; 2^1 \; 2^2 \; 2^3 \; 2^4 \; 2^5 \; 2^x$

$\underbrace{j = 1, 2, 4, 8, 16, 32 \dots n}$ # steps $= x$

```
j = 1;
while j <= n do
    i = 1;
    while i <= n do
        i = i + j;
    end while
    j++;
end while
```

while $j <= n$ do — $(n+1)$ time

$n+1$

$\frac{n}{j}$ time

$j = 1$; $i = 1, 2, 3, 4, \dots n = n$ times

$j = 2$; $i = 1, 3, 5, 7, \dots n = n/2$ times

$+2 \quad +2 \quad +2$

$j = 3$; $i = 1, 4, 7, 10, \dots n = n/3$ times

$+3 \quad +3 \quad +3$

} $\frac{n}{j}$ times

$\Rightarrow$ Total # times in inner loop

$$\Rightarrow \sum_{j=1}^{n} \frac{n}{j}$$

$$\therefore \sum_{i=1}^{n} \frac{1}{i} = \Theta(\log n) \text{ "nth harmonic \#"}$$

$$= n \sum_{j=1}^{n} \frac{1}{j} = n \cdot \log n \Rightarrow \Theta(n \log n)$$

$$\therefore \sum_{i=a}^{n} x = \sum_{i=1}^{n} x - \sum_{i=1}^{a} x$$

// now sum starts at 1

Summations:

(1) $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ (arithmetic series)

(2) $\sum_{i=1}^{n} \frac{1}{i} \in \Theta(\log n)$ (n-th harmonic number)

(3) $\sum_{i=0}^{n} a^i = \frac{a^{n+1}-1}{a-1}$ (geometric series) a is constant, $a \neq 1$

(4) $\sum_{i=0}^{\infty} a^i = \frac{1}{1-a}$ (infinite geometric series) $a < 1$

i.e. $\left(\frac{1}{2}\right)^i$ $a = 1/2$

Proof of arithmetic series:

Base Case: $n = 1$

$$\Rightarrow \sum_{i=1}^{1} i = \frac{1(1+1)}{2} = 1$$

(LHS) (RHS)

Inductive Step: k

$$\Rightarrow \text{Assume: } \sum_{i=1}^{k} i = \frac{k(k+1)}{2}$$

$$\text{Prove: } \sum_{i=1}^{k+1} i = \frac{k+1(k+1+1)}{2} = \frac{(k+1)(k+2)}{2}$$

$$\Rightarrow \sum_{i=1}^{k+1} i = (k+1) + \underbrace{\sum_{i=1}^{k} i}_{\text{Arithmetic}} = (k+1) \cdot \frac{2}{2} + \frac{k(k+1)}{2}$$

$\leftarrow$ Inductive hypothesis

$$= \frac{2(\cancel{k+1}) + k(\cancel{k+1})}{2}$$

// factor (k+1)

$$= \frac{(k+1)(k+2)}{2} = \text{RHS} \quad \square$$

$\therefore$ goal is to get this to look like assumption