Asymptotic lower bound

Definition [Omega] $f(n) = \Omega(g(n))$ iff there exist positive constants c and n_0 such that $f(n) \geq c\,g(n)$ for all $n \geq n_0$.

Example

$3n+2 = \Omega(n)$ as $3n+2 \geq 3n$ for $n \geq 1$

$6 \cdot 2^n + n^2 = \Omega(2^n)$ as $6 \cdot 2^n + n^2 \geq 6 \cdot 2^n$, $n \geq$

$10n^4 + 4n + 2 = \Omega(n^4)$ as $\boxed{} \geq n^4$, $n \geq$

Observe also that

$3n+3 = \Omega(1)$

$10n^4 + 4n = \Omega(n)$ $\left(= \Omega(1)\right)$

$6 \cdot 2^n + n^2 = \Omega(n^2)$

The rule: take $g(n)$ as large as possible

Thm If $f(n) = a_m n^m + \cdots + a_1 n + a_0$ $a_m \neq 0$ and $a_m > 0$ then $f(n) = \Omega(n^m)$

$n^2 - mn$

Asymptotic tight bounds

Definition [Theta] $f(n) = \Theta(g(n))$ iff there exist positive constants c_1, c_2 and n_0 such that $c_1 g(n) \leq f(n) \leq c_2 g(n)$ for all $n \geq n_0$

[$f(n)$ can be "sandwiched" between $c_1 g(n)$ and $c_2 g(n)$]

Note that the statement $f(n) = O(g(n))$ that $g(n)$ is an <u>asymptotic upper bound</u> on the value of $f(n)$. Notice that

$n = O(n)$ $\quad n = O(n^2)$ $\quad n = O(n^{2.5})$ ..

<u>The rule</u>: take $g(n)$ as small as a function of n as possible as one can come up with for which $f(n) = O(g(n))$

So we say that $3n+3 = O(n)$, we shall almost never say that $3n+3 = O(n^2)$ even this latter statement is correct.

<u>Running time classification</u>

$O(1)$ — constant running time

$O(n)$ — linear

$O(n^2)$ — ~~cubic~~ quadratic

$O(n^3)$ — ~~qu~~ cubic (in general polynomial)

$O(2^n)$ — exponential.

<u>Thm</u> If $f(n) = a_m n^m + a_{m-1} n^{m-1} + \dots + a_1 n + a$
$a_i \geq 0$, then $f(n) = O(n^m)$

<u>Proof</u> $f(n) = n^m \left(a_m + a_{m-1} \frac{n^{m-1}}{n^m} + \right.$

$$\left. + a_{m-2} \frac{n^{m-2}}{n^m} + \dots + a_1 \frac{n}{n^m} + a_0 \cdot \frac{1}{n^m} \right) \leq$$

$$\leq n^m \sum_{j=0}^{m} a_j = c n^m$$

∎

Example: $1+2+\dots+n = O(n^2)$

Example

$3n+2 = \Theta(n)$ as $3n+2 \geq 3n$ and
$3n+2 \leq 4n$ for all $n \geq 2$

$10n^2+4n+2 = \Theta(n^2)$

$6 \cdot 2^n + n^2 = \Theta(2^n)$

Notice however

$10n^2+4n+2 = O(n^3)$

but $10n^2+4n+2 \neq \Theta(n^3)$

since you can not sandwiched it in the form

no $c_1 n^3 \leq 10n^2+4+2 \leq c_2 n^3$ ✓ ok

$c_1, c_2, \forall n \geq n_0$

Similarly

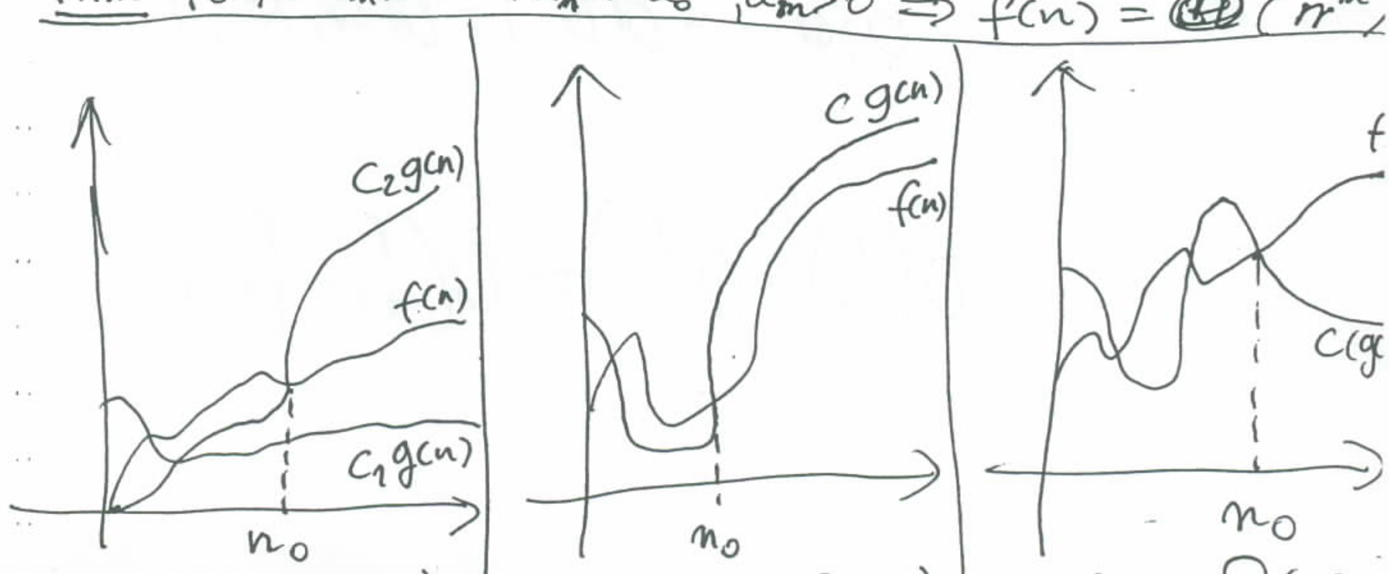
$10n^2+4n+2 = \Omega(n)$

but $10n^2+4n+2 \neq \Theta(n)$ since

$c_1 n \leq 10n^2+4+2 \leq c_2 n$

no c_1, c_2 s.t. $\forall n \geq n_0$

Thm $f(n) = a_m n^m + \dots + a_1 n + a_0$, $a_m > 0 \Rightarrow f(n) = \Theta(n^m)$



$f(n) = \Theta(g(n))$ $f(n) = O(g(n))$ $f(n) = \Omega(g(n))$

9.3 O MANIPULATION

Like any mathematical formalism, the O-notation has rules of manipulation that free us from the grungy details of its definition. Once we prove that the rules are correct, using the definition, we can henceforth work on a higher plane and forget about actually verifying that one set of functions is contained in another. We don't even need to calculate the constants C that are implied by each O, as long as we follow rules that guarantee the existence of such constants.

The secret of being a bore is to tell everything.
— *Voltaire*

For example, we can prove once and for all that

$$n^m = O(n^{m'}), \qquad \text{when } m \leqslant m'; \tag{9.21}$$

$$O(f(n)) + O(g(n)) = O(|f(n)| + |g(n)|). \tag{9.22}$$

$O(f_1)+O(f_2)$

Then we can say immediately that $\frac{1}{3}n^3+\frac{1}{2}n^2+\frac{1}{6}n = O(n^3)+O(n^3)+O(n^3) = O(n^3)$, without the laborious calculations in the previous section.

Here are some more rules that follow easily from the definition:

$$\begin{aligned}
f(n) &= O(f(n)); && (9.23)\\
c\cdot O(f(n)) &= O(f(n)), \qquad \text{if } c \text{ is constant}; && (9.24)\\
O(O(f(n))) &= O(f(n)); && (9.25)\\
O(f(n))O(g(n)) &= O(f(n)g(n)); && (9.26)\\
O(f(n)\,g(n)) &= f(n)O(g(n)). && (9.27)
\end{aligned}$$

Exercise 9 proves (9.22), and the proofs of the others are similar. We can always replace something of the form on the left by what's on the right, regardless of the side conditions on the variable n.

Equations (9.27) and (9.23) allow us to derive the identity $O(f(n)^2) = O(f(n))^2$. This sometimes helps avoid parentheses, since we can write

$$O(\log n)^2 \qquad \text{instead of} \qquad O((\log n)^2).$$

Both of these are preferable to '$O(\log^2 n)$', which is ambiguous because some authors use it to mean '$O(\log\log n)$'.

Can we also write

$$O(\log n)^{-1} \qquad \text{instead of} \qquad O((\log n)^{-1})\,?$$

No! This is an abuse of notation, since the set of functions $1/O(\log n)$ is neither a subset nor a superset of $O(1/\log n)$. We could legitimately substitute $\Omega(\log n)^{-1}$ for $O((\log n)^{-1})$, but this would be awkward. So we'll restrict our use of "exponents outside the O" to cases where the exponents are positive.

9.3 O MANIPULATION

Like any mathematical formalism, the O-notation has rules of manipulation that free us from the grungy details of its definition.

For example. we can prove once and for all that

$$n^m = O(n^{m'}), \qquad \text{when } m \le m'; \tag{9.21}$$

f_1 τ_1

$$h = \overset{f_1}{O(f(n))} + O(g(n)) = O(\,|f(n)| + |g(n)|). \tag{9.22}$$

$= O(\max\{f, g\})$

$O(\max(f(n), g(n)))$

Here are some more rules that follow easily from the definition:

$$
\begin{aligned}
f(n) &= O(f(n)); && (9.23)\\
c \cdot O(f(n)) &= O(f(n)), \quad \text{for } c \text{ is constant}; && (9.24)\\
O(O(f(n))) &= O(f(n)); && (9.25)\\
O(f(n))O(g(n)) &= O(f(n)g(n)); && (9.26)\\
\cancel{O(f(n)g(n))} &= \cancel{f(n)O(g(n))}). && (9.27)
\end{aligned}
$$

$h_1 = O(f)$ proof $f_1 = O(f(n))$ $g_1 = O(g(n))$

$h_2 = O(g)$

$h_1(n)$ $|f_1(n)| \le c_1 |f(n)|$ $n \ge n_{01}$

$h_1 + h_2 = O(f+g)$

$h_2(n)$ $|f_2(n)| \le c_2\, g(n)$ $n \ge n_{02}$

$f_1(n) + f_2(n) \le c_1 f(n) + c_2 g(n) \le \max(c_1, c_2)\{f(n) + g(n)\}$

f $n \ge \max\{n_{01}$

9.1. A HIERARCHY

Functions of n that occur in practice usually have different "asymptotic growth ratios"; one of them will approach infinity faster than another. We formalize this by saying that

$f(n) = o(g(n))$

$$f(n) \prec g(n) \iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0. \qquad (9.3)$$

This relation is transitive: If $f(n) \prec g(n)$ and $g(n) \prec h(n)$ then $f(n) \prec h(n)$. For example, $n \prec n^2$; informally we say that n grows more slowly than n^2.

There are, of course, many functions of n besides powers of n.

$$\boxed{1 \prec \log\log n \prec \log n \prec n^{\epsilon} \prec n^{c} \prec n^{\log n} \prec c^{n} \prec n^{n} \prec c^{c^{n}}.}$$

(Here $\in$ and c are arbitrary constants with $0 < \in < 1 < c$.)

All functions listed here, except 1, go to infinity as n goes to infinity. Thus when we try to place a new function in this hierarchy, we're not trying to determine *whether* it becomes infinite but rather *how fast*.

Examples

(1) $\sum_{i=1}^{n} i = \Theta(n^2)$ because $\sum_{i=1}^{n} i = \frac{n(n+1)}{2} = \frac{n^2}{2} + O(n)$

$\sum_{i=1}^{n} i^2 = \Theta(n^3)$ because

Upper bound: $\sum i^2 \le n^2 \cdot \sum 1 = n^3$

Lower bound:

$$\sum_{i=1}^{n} i^2 \ge \sum_{i=n/2}^{n} i^2 \ge \left(\frac{n}{2}\right)^2 \cdot \frac{n}{2} = \frac{n^3}{8}$$

---

(2) $n!$

Some bounds,

$$n! = 1 \cdot 2 \cdots n \le n^n$$

$$n! = 1 \cdot 2 \cdots \left(\frac{n}{2}\right) \cdots n \ge \left(\frac{n}{2}\right)^{n/2}$$

Stirling formula

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n\left(1 + O\left(\frac{1}{n}\right)\right)$$

Examples:

Some Taylor expansions

$$e^x = 1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + O(x^4) \qquad x \to 0$$

$$\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} + O(x^4)$$

$$\frac{1}{1-x} = 1 + x + x^2 + x^3 + O(x^4) \qquad \sum x^n = \frac{1}{1-x}$$

$$(1+x)^\alpha = 1 + \alpha x + \binom{\alpha}{2} x^2 + O(x^3) \qquad \text{Newton}$$

Two examples in $n!$

Example 1: Evaluate $(n-1)!$

$$(n-1)! = \sqrt{2\pi(n-1)}\left(\frac{n-1}{e}\right)^{n-1}$$

$$\sqrt{n-1} = \sqrt{n}\left(1-\frac{1}{n}\right)^{\frac{1}{2}} = \sqrt{n}\left(1 - \frac{1}{2n} + O\left(\frac{1}{n^2}\right)\right) = \sqrt{n}\left(1 - \frac{1}{2n} + O\left(\frac{1}{n^2}\right)\right)$$

$$(n-1)^{n-1} = n^{n-1}\left(1-\frac{1}{n}\right)^{n-1} = n^{n-1}\left(1-\frac{1}{n}\right)^{n}\frac{1}{1-\frac{1}{n}} = n^{n-1}\left(1-\frac{1}{n}\right)^{n}(1+\ldots)$$

$$\left(1-\frac{1}{n}\right)^{n} = \exp\left[n \ln\left(1-\frac{1}{n}\right)\right]$$

$$= \exp\left[(n)\left(-\frac{1}{n} - \frac{1}{2n^2} + O\left(\frac{1}{n^3}\right)\right)\right] =$$

$$= \exp\left(-1 - \frac{1}{2n} + O\left(\frac{1}{n^2}\right)\right) = e^{-1}\exp\left(-\frac{1}{2n}\right)\exp\left(O\left(\frac{1}{n^2}\right)\right)$$

$$= e^{-1}\left(1 - O\left(\frac{1}{n}\right)\right)$$

Thus,

$$(n-1)! = \sqrt{2\pi n}\left(1-O\left(\tfrac{1}{n}\right)\right) \cdot \frac{n^{n-1}}{e^{n-1}} \cdot \frac{1}{e}\left(1+O\left(\tfrac{1}{n}\right)\right)$$

$$= \sqrt{2\pi n} \cdot \left(\frac{n}{e}\right)^{n-1} \cdot \frac{1}{e}\left(1+O\left(\tfrac{1}{n}\right)\right)$$

Example 2: Harold

$$n\left(\sqrt[n]{n}-1\right) \sim \ ?$$

$$\sqrt[n]{n} = n^{\frac{1}{n}} = \exp\left(\tfrac{1}{n}\ln n\right) = 1 + \frac{\ln n}{n} + O\left(\frac{\ln^2 n}{n^2}\right)$$

$$\left(\sqrt[n]{n}-1\right) = \frac{\ln n}{n} + O\left(\frac{\ln^2 n}{n^2}\right)$$

$$n\left(\sqrt[n]{n}-1\right) = \ln n + O\left(\frac{\ln^2 n}{n}\right)$$

What is the runtime of this code segment in big-O notation?

5. We write an alternate program for the same problem. In this case, we notice that to compute `output_list[i]`, you only need to add `input_list[i]` and `output_list[i-1]`. The program segment is as follows:

```
for (i = 0; i < n; i++)
    output_list[i] = 0;
output_list[0] = input_list[0];
for (i = 1; i < n; i++)
    output_list[i] = output_list[i-1] + input_list[i];
```

What is the runtime of this code segment in big-O notation?

6. Code the programs for computing cumulative sum of a list (illustrated in Problems 4 and 5) in Java. Execute each of these programs on lists of increasing sizes and note the runtime. Plot the runtime of the two programs on a graph. Fit a degree two polynomial curve through each of these plots and note down the coefficients. What can you say about the coefficients and the time for the primitive operations in the two programs?

Project Notes.

Identify the various components of their selected project problem, interrelationship between components, and prepare a preliminary design and requirements specification for each of the components.