How to show that a function is not big-oh of another function?

Example: Show that $n^2 \neq O(n)$.

Solution: We need to show that there are no constants $C, K > 0$ such that

$$n^2 \leq Cn$$

for every $n \geq K$.

That is, for every pair of constants $C, K > 0$, we need to find a value of $n \geq K$ such that $n^2 > Cn$.

Take $n > \max\{C, K\}$. Then,

$$n^2 = n \cdot n > Cn,$$

as we want it.

□

Comparing the growth of important functions

Theorem: Let $f(x) = a_n x^n + \dots + a_1 x + a_0$, where $a_n, \dots, a_1, a_0 \in \mathbb{R}$. Then, $f(x) = O(x^n)$.

To prove this theorem, we will use the triangular inequality.

Lemma (Triangular inequality) Let $x, y \in \mathbb{R}$. We have $|x+y| \leq |x| + |y|$.

Proof of the lemma: Let $x, y \in \mathbb{R}$.

Observe that $xy \leq |x||y|$. and $|x|^2 = x^2$ and $|y|^2 = x^2$. Thus, we have

$$x^2 + 2xy + y^2 \leq |x|^2 + 2|x||y| + |y|^2$$

$$(x+y)^2 \leq (|x| + |y|)^2.$$

This implies $|x+y| \leq \big||x| + |y|\big| = |x| + |y|$ ∎

Now we can prove our theorem:

Proof of theorem:

Note that

$$|f(x)| = |a_n x^n + b_{n-1} x^{n-1} + \dots + a_1 x + a_0|$$

$$\leq |a_n x^n| + \dots + |a_1 x| + |a_0|$$

↑ Triangular inequality

$$\leq |x^n| \cdot \left(|a_n| + \left|\frac{a_{n-1}}{x}\right| + \dots + \left|\frac{a_1}{x^{n-1}}\right| + \left|\frac{a_0}{x^n}\right| \right)$$

When $x > 1$, we have that

$$\left| \frac{a_i}{x^{n-i}} \right| \leq |a_i|$$

for every $i = 0, \ldots, n$.

We conclude that

$$|f(x)| \leq |x^n| \cdot (|a_n| + \ldots + |a_1| + |a_0|)$$

for every $x \geq 1$. Taking $|a_n| + \ldots + |a_0|$ as our constant, we conclude our proof ∎

Observation:

We also have that $x^n = O(a_n x^n + \ldots + a_1 x + a_0)$.

We leave the proof as an exercise.

The factorial function is defined as

$$n! = 1 \cdot 2 \cdot 3 \cdots n$$

(multiplication of the 1st n natural numbers).

We also define $0! = 1$.

The main functions used to measure time complexity are: polynomials, exponentials and logarithms. Now let us recall the definition of exponential and logarithm.

Definition (exponential function).

An exponential function is a function $f: \mathbb{R} \to \mathbb{R}_{>0}$ of the form $f(x) = b^x$, where $b > 0$.
The number b is called base.

Definition (logarithm function)

An logarithm function is the inverse of an exponential function.
We denote by $\log_b : \mathbb{R}_{>0} \to \mathbb{R}$ the inverse function of $f(x) = b^x$.
We have $\log_b(x) = a$ if $b^a = x$.

Time complexity

The time complexity of an algorithm can be expressed in terms of the number of operations used by the algorithm when the input has a particular size.

Operations used to measure time:

comparison of integers, addition, multiplication, division or any other basic operation.

Example: Use the number of comparisons to measure the time complexity of the following algorithm:

Algorithm to find the maximum element in a finite seq:

PROCEDURE max ($a_1, \cdots, a_n$: real numbers)

max $\leftarrow a_1$

For $i \in [2, n]$:

    if max $< a_i$, then max $\leftarrow a_i$

RETURN max.

At each iteration of the FOR loop, two comparisons are made: one to see if we should exit the loop ($i \leq n$) and other for max $< a_i$.
One more is used when we exit the loop.
In total, we have $2(n-1)+1$ comparisons.
Time complexity: $O(n)$.

Example: Use the number of comparisons to measure the time complexity of the following algorithm:

Linear search algorithm

PROCEDURE linear search ($x \in \mathbb{R}, (a_1, \cdots, a_n) \in \mathbb{R}^n$)

$i \leftarrow 1$

WHILE $i \leq n$ and $x \neq a_i$:

    $i \leftarrow i+1$

If $i \leq n$, then location $\leftarrow i$

Else location ← 0

RETURN location.

At each iteration of the loop, two comparisons are made: $i \leq n$ and $x \neq a_i$.

One more is used when we exit the loop.

Another is used outside the loop.

In total, we have $2n + 2$ comparisons.

Time complexity: $O(n)$.